Российская ассоциация искусственного интеллекта
Санкт-Петербургский институт информатики и автоматизации
Российской академии наук

ПЯТЫЙ ВСЕРОССИЙСКИЙ НАУЧНО-ПРАКТИЧЕСКИЙ СЕМИНАР «БЕСПИЛОТНЫЕ ТРАНСПОРТНЫЕ СРЕДСТВА С ЭЛЕМЕНТАМИ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА» (БТС-ИИ-2019)

22-24 мая 2019
г. Санкт-Петербург, Россия

Труды семинара

УДК 004.8
ББК-32.813
П99

Пятый Всероссийский научно-практический семинар «Беспилотные транспортные средства с элементами искусственного интеллекта» (БТС-ИИ-2019): Труды семинара. – Переславль-Залесский: Российская ассоциация искусственного интеллекта, 2019. – 264 с.

ISBN 978-5-6042802-0-1

В сборник включены тексты работ, представленные на пятом Всероссийском научно-практическом семинаре «Беспилотные транспортные средства с элементами искусственного интеллекта» (БТС-ИИ-2019) 22-24 мая 2019 года.

Официальный сайт семинара – www.ai-uv.ru

ISBN 978-5-6042802-0-1

О семинаре

В настоящее время наблюдается существенное повышение интереса исследователей и разработчиков к созданию беспилотных транспортных средств различного типа и назначения, способных к автономному решению высокоуровневых задач в динамических, непрогнозируемых средах. Создание подобных средств невозможно без интеграции усилий специалистов в различных областях науки: механики, теории управления, теории передачи информации, компьютерной графики, распознавания образов, искусственного интеллекта, когнитивных наук и многих других. Одним из механизмов указанной интеграции является проведение семинара «Беспилотные транспортные средства с элементами искусственного интеллекта» (БТС-ИИ), в ходе которого исследователи различных специализаций имеют возможность обменяться опытом решения актуальных проблем, связанных с созданием беспилотных транспортных средств нового поколения – таких средств, которые могли бы быть названы интеллектуальными.

Первый семинар БТС-ИИ состоялся в 2014 году в Казани в рамках Четырнадцатой национальной конференции по искусственному интеллекту с международным участием (КИИ-2014). Со следующего года семинар проводится как самостоятельное мероприятие, а в 2019 году – состоится уже в пятый раз. В этом году площадкой семинара выступает Санкт-Петербургский институт информатики и автоматизации Российской академии наук, известный своими успехами и достижениями в области интеллектуальных систем, мобильной робототехники и пр. На семинар поступила 31 работа, принятно по итогам рецензирования – 24.

Желаем успехов участникам семинара и надеемся на плодотворную дискуссию в ходе работы!

Программный комитет БТС-ИИ-2019

Организатор

Российская ассоциация искусственного интеллекта (www.raai.org)

Санкт-Петербургский институт информатики и автоматизации Российской академии наук (www.spiiras.nw.ru)

Программный комитет

В.Е. Павловский (сопредседатель), доктор физико-математических наук, профессор, главный научный сотрудник Федерального исследовательского центра "Институт прикладной математики им. М.В. Келдыша" Российской академии наук, член научного совета Российской ассоциации искусственного интеллекта.

А.Л. Ронжин (сопредседатель), доктор технический наук, профессор, директор Санкт-Петербургского института информатики и автоматизации Российской академии наук.

К.С. Яковлев (сопредседатель), кандидат физико-математических наук, ведущий научный сотрудник Федерального исследовательского центра "Информатика и управление" Российской академии наук, член Российской ассоциации искусственного интеллекта.

И.М. Афанасьев, кандидат технических наук, доцент лаборатории программной инженерии, институт разработки ПО и программной инженерии, Университет Иннополис.

Д.А. Добрынин, кандидат технических наук, старший научный сотрудник Федерального исследовательского центра "Информатика и управление" Российской академии наук, член Российской ассоциации искусственного интеллекта.

В.Э. Карпов, кандидат технических наук, доцент, начальник лаборатории робототехники Национального исследовательского центра "Курчатовский институт", вице-президент Российской ассоциации искусственного интеллекта

Н.В. Ким, кандидат технических наук, профессор, лауреат премии правительства РФ в области образования, профессор кафедры 704 факультета №7 "Робототехнические и интеллектуальные системы" Московского авиационного института.

Е.А. Магид, PhD, профессор, Senior IEEE member, заведующий кафедрой интеллектуальной робототехники, Высшая школа информационных технологий и интеллектуальных систем (ИТИС), Казанский (Приволжский) федеральный университет

Организационный комитет

А.И. Савельев (сопредседатель), кандидат технических наук, заведующий лабораторией автономных робототехнических систем, Санкт-Петербургского института информатики и автоматизации Российской академии наук.

К.С. Яковлев (сопредседатель), кандидат физико-математических наук, ведущий научный сотрудник Федерального исследовательского центра "Информатика и управление" Российской академии наук, член научного совета Российской ассоциации искусственного интеллекта.

А.А. Андрейчук, аспирант РУДН

А.В. Боковой, аспирант РУДН, м.н.с. ФИЦ ИУ РАН

И.В. Ватаманюк, м.н.с., аспирант лаборатории автономных робототехнических систем СПИИРАН

Д.А. Малов, м.н.с., аспирант лаборатории автономных робототехнических систем СПИИРАН

П.М. Черноусова, магистр, специалист 2-й категории ОИГ ИШИВ ГУАП

Официальный сайт семинара

www.ai-uv.ru

Содержание

УДК 004.896:621.865

НЕДООПРЕДЕЛЁННЫЕ МОДЕЛИ В ЗАДАЧЕ ЛОКАЛИЗАЦИИ МОБИЛЬНОГО РОБОТА

А.Д. Московский (*moscowskyad@gmail.com*)
Национальный исследовательский центр
«Курчатовский институт», Москва

Аннотация. В работе исследуется возможность применения метода недоопределённых вычислений для задачи определения мобильным роботом своего положения. Рассматривается классическая проблема локализации по данным одометрии и наблюдаемым при помощи камеры ориентирам. Особенностью рассматриваемой задачи является наличие зон неопределённости, в которых отсутствуют данные об ориентирах. Определены преимущества разрабатываемого метода на основе недоопределённых вычислений в применении к рассматриваемому классу задач в сравнении с классическими методами локализации. Разработанная система локализации сравнивается с методами на основе расширенного фильтра Калмана и фильтра частиц. Проведенные вычислительные эксперименты показали преимущество разрабатываемой системы на определенном классе задач.[1]

Ключевые слова: локализация, мобильный робот, недоопределённые вычисления.

Введение и постановка задачи

Задача локализации робота в пространстве возникла с момента зарождения мобильной робототехники, т.к. любая задача, связанная с движением, неминуемо требует ответа на вопрос «где я?». Классической задачей можно назвать задачу, когда робот совершает движение в среде, основываясь на данных о своем перемещении, и корректирует свое положение, ориентируясь на расположенные в его окружении метки. Предполагается, что положения данных меток в среде известны, а также возможно определить расстояние до них или угол наблюдения. Метки могут быть объектами весьма широкого класса, например, специальные радиомаяки. Однако наиболее удобными в использовании являются

[1] Работа выполнена при поддержке НИЦ «Курчатовский институт» (приказ от 05.07.2018 №1601).

визуальные ориентиры, т.к. веб-камера является одним из самых доступных датчиков, а также в качестве ориентиров зачастую можно использовать уже расположенные в среде функционирования объекты. Первая проблема такой задачи - это погрешности измерений, которые касаются как определения своего смещения по внутренним датчикам, так и измерений, связанных с определением параметров меток. Вторая проблема связана с зонами неопределённости, в которых робот не наблюдает ни одного ориентира. Если первая проблема неизбежна, то вторая часто не рассматривается, т.к. исследователи обычно пытаются поместить в среду достаточное количество ориентиров. В данной же работе такие случаи рассматриваются отдельно.

1. Классические методы локализации

Основным столпом, на котором основано большинство использующихся методов навигации, являются вероятностные подходы. Они подробно описаны в монографии Себастьяна Трана [Thrun и др., 2005]. К вероятностным подходам относятся такие известные методы, как фильтр Калмана и фильтр частиц. Они используются для того, чтобы бороться с неопределённостью, которая возникает ввиду таких факторов, как: неструктурированность среды оперирования; зашумленность сенсоров робота; неточности исполняемых механизмов (актуаторов). Обычно такие подходы работают по схеме «предсказание-коррекция». На каждом шаге при помощи модели системы вычисляется новое положение робота, которое потом корректируется в соответствии с данными от сенсоров робота. Далее будут рассмотрены самые часто используемые на практике подходы: фильтр Калмана и фильтр частиц.

1.1 Фильтр Калмана

Существует много реализаций фильтра Калмана, в робототехнике в основном используется расширенный фильтр Калмана (Extended Kalman Filter) [Einicke и др., 1999], который будет рассмотрен далее. EKF строится на знании о модели движения робота и её погрешности, а также о доступных сенсорных измерениях и их ошибках. Этап предсказания на каждом шаге работы, используя модель движения робота, определяет его смещение и выдвигает гипотезу о новом положении робота. Эта гипотеза также содержит компоненту погрешности. На этом же этапе вычисляется ковариационная матрица для экстраполированного вектора положения. Далее на этапе коррекции рассчитываются отклонения показаний сенсоров от ожидаемых экстраполированных значений для получения матрицы усиления, на основе которой вычисляется коррекция. Положение робота рассчитываются как предсказанное положение робота, сложенное с рассчитанной коррекцией. Матрица коэффициентов также обновляется.

Несмотря на свою распространённость, EKF имеет ряд недостатков: требуется знать модель робота с достаточно хорошей приближенностью; обязательна информация о погрешностях измерений; требуется серьезная перенастройка при меняющихся измеряемых данных; нелинейные задачи требуют перестроения матриц моделей на каждом шаге алгоритма.

1.2. Фильтр частиц

Впервые понятие фильтр частиц было озвучено в 1996 году [Moral Del, 1996], в русскоязычной литературе еще встречается название многочастичный фильтр. Фильтр частиц относят к методам семейства Монте-Карло. Стартовая область заполняется случайным образом «частицами», которые являются моделями робота. Между частицами равномерно распределятся вес, в совокупности равный единице. Соответственно, когда робот совершает движение, совершает движение и каждая частица, меняя свое положение. Когда поступает информация, позволяющая уточнить свое положение, вес каждой частицы пересчитывается. Пересчет веса происходит обычно с использованием распределения Гаусса. Чем больше отдельно взятая частица удовлетворяет пришедшим данным, тем больше её вес. После перераспределения весов и их нормировки происходит главный «трюк» фильтра частиц, называющийся в англоязычной литературе resampling. В результате отбрасываются частицы с наименьшим весом, а частицы с большим весом копируются до изначального числа. Таким образом, в ходе нескольких итераций остаются только частицы в области истинного положения робота. Финальное положение робота вычисляется как сумма положений частиц, помноженных на свой вес.

В сравнении с фильтром Калмана фильтр частиц лучше справляется с нелинейными задачами. Однако среди недостатков данного метода можно отметить то, что они более «тяжелые» по вычислениям, ибо корректность фильтрации улучшается с ростом количества частиц. Поскольку инициализация фильтра и resampling выполняются при помощи генератора случайных чисел, это может сильно повлиять на стабильность метода. Также в случае если фильтр «пошел по ложному следу», ему очень сложно перестроиться на истинный путь.

2 Предлагаемый подход

2.1 Общие понятия в недоопределенных вычислениях

Недоопределённые модели (далее н-модели) были впервые предложены отечественным выдающимся ученым Александром Семеновичем Нариньяни в 1986 [Нариньяни, 1986]. Основная идея данного подхода заключается в том, чтобы не описывать всю исследуемую систему целиком,

а лишь её некоторую модель, которая задается набором переменных и функций. Также вводится понятие недоопределённой переменной (далее н-переменной), которая представляет собой некоторую область, в которой находится истинное значение переменной в её классическом понимании. Область может быть представлена как интервалом или мультинтервалом, так и дискретным списком. Н-переменные связаны между собой функциями присваивания (иногда встречаются как функции интерпретации). Модель представима в виде двудольного графа, где один набор вершин это Н-переменные, а другой - функции присваивания. Когда в систему попадают новые данные извне, то они изменяют Н-переменные, которые в свою очередь активируют функции, для которых они являются входными аргументами, функции изменяют другие Н-переменные и т.д. В ходе вычислительного процесса область неопределённости переменных сужается и должна сходиться к своему точному значению.

Формально обобщенная вычислительная модель задаётся следующей четверкой [Нариньяни, 2007]:

$$M = (V, R, W, C), \quad (1)$$

где V – множество параметров из заданной области, R – множество ограничений, W – множество функций присваивания, C – множество функций проверки корректности. Н-вычисления позволяют решать широкий круг задач, в т. ч. применялись для задачи навигации мобильного робота по набору маяков с известным направлениями на них [Карпов, 2009].

2.2 Недоопределенные вычисления для задачи локализации

Предложенный метод также относится к классу методов «предсказание-коррекция» и параллельно рассчитывает положение робота, определяемое его координатами, и ограничивающую область, заданную н-переменными. Ограничивающая область - это область, в которой потенциально может находиться робот, основываясь на начальных и входных данных. При поступлении данных о перемещении положение робота и ограничивающая область изменяются. Далее на основе данных о наблюдаемых ориентирах ограничивающая область изменяется, и если измененное положение робота выходит за переделы ограничивающей области, то оно корректируется на максимально близкое ему значение, принадлежащее области ограничения.

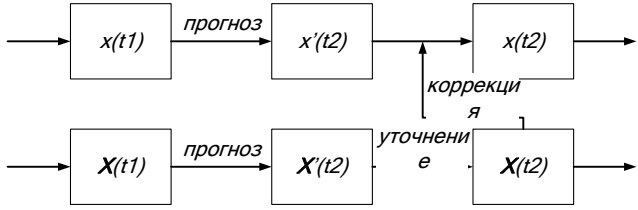


Рис. 1. Схема работы метода локализации на основе н-моделей

В двумерном случае положение робота x_r задается тремя переменными: координатами на плоскости *(x, y)* и углом поворота α. Ограничивающая область X_r задается тремя Н-переменными: координата $*x$, координата $*y$ и угол поворота робота $*\alpha$. Каждая Н-переменная представляет собой непрерывный интервал от минимально возможного значения до максимально возможного значения.

$$X_r = \begin{pmatrix} x_{min} & x_{max} \\ y_{min} & y_{max} \\ \alpha_{min} & \alpha_{max} \end{pmatrix} \quad (2)$$

На старте метода положение робота должно быть инициализировано либо его примерным положением, либо, если таковое не известно, то всей зоной, в которой оперирует робот. Ориентация робота в отсутствии знаний о стартовом положении задается от $-\pi$ до π.

Данные о смещении робота за момент времени t_2-t_1 представляют собой линейное dr и угловое $d\alpha$ смещения. При смещении x_r изменяется следующим образом:

$$x_r' = (x + dr \cdot cos(\alpha + d\alpha), y + dr \cdot sin(\alpha + d\alpha), \alpha + d\alpha) \quad (3)$$

Когда совершается движение, предполагается, что робот перемещается в области, заданной смещением, измененным на относительную погрешность измерений δ. Т.к. угол задается в некотором диапазоне значений, то область расширяется на линейное смещение, изменённое на погрешность измерений, в каждом направлении, принадлежащему диапазону углов.

$$X_r' = \bigcup_{\alpha = \alpha_{min}(1-\delta_\alpha)}^{\alpha_{max}(1+\delta_\alpha)} X_r + \begin{pmatrix} min\big(dr \cdot cos(\alpha) \cdot (1 \pm \delta_r)\big) & max(dr \cdot cos(\alpha) \cdot (1 \pm \delta_r)) \\ min\big(dr \cdot sin(\alpha) \cdot (1 \pm \delta_r)\big) & max(dr \cdot sin(\alpha) \cdot (1 \pm \delta_r)) \\ min\big(d\alpha \cdot (1 \pm \delta_\alpha)\big) & max(d\alpha \cdot (1 \pm \delta_\alpha)) \end{pmatrix} \quad (4)$$

Согласно обобщенной вычислительной модели, данная операция «раздвижения» области является функцией присваивания, обозначим её w^+.

На следующем этапе происходит процесс уточнения. В общем случае уточняющими данными может быть практически любая информация, получаемая от датчиков робота, которая каким-либо образом влияет на X_r. В данном случае происходит уточнение положения по данным маяков *O*, координаты которых известны, а расстояние до них определяется с известной погрешностью. Когда маяк попадает в «поле зрения» робота, то область максимального удаления от маяка X_o можно определить как квадрат со стороной, равной удвоенному расстоянию от маяка *d*, увеличенному на погрешность измерений δ_d.

$$X_o = \begin{pmatrix} x_o - d(1+\delta_d) & x_o + d(1+\delta_d) \\ y_o - d(1+\delta_d) & y_o + d(1+\delta_d) \\ -\pi & \pi \end{pmatrix} \quad (5)$$

Соответственно, робот должен находиться как в области X_r так и в области X_o, а это значит, что он находится на пересечении данных областей, если таковое существует. Для определения области пересечения также требуется определить соответствующую функцию присваивания $w^\times$ *из* обобщенной вычислительной модели (1). Если пересечение областей существует, то функция присваивания $w^\times$ его определяет, в Н-вычислениях это называется сужением области неопределённости. В противном случае берется объединение областей, т.к. неопределённость не удалось сузить. Аналогично происходит и с диапазоном углов. После данной операции Н-переменные могут представлять собой мультиинтервалы.

$$w^\times(X_1, X_2) = \begin{cases} X_1 \cap X_2, & X_1 \cap X_2 \neq \emptyset \\ X_1 \cup X_2, & X_1 \cap X_2 = \emptyset \end{cases} \quad (6)$$

$$X_1 \cap X_2 = \begin{pmatrix} max(x^1_{min}, x^2_{min}) & min(x^1_{max}, x^2_{max}) \\ max(y^1_{min}, y^2_{min}) & min(y^1_{max}, y^2_{max}) \\ max(\alpha^1_{min}, \alpha^2_{min}) & min(\alpha^1_{max}, \alpha^2_{max}) \end{pmatrix} \quad (7)$$

$$X = \emptyset \Leftarrow \begin{cases} x_{min} > x_{max} \\ y_{min} > y_{max} \\ \alpha_{min} > \alpha_{max} \end{cases} \quad (8)$$

$$X_1 \cup X_2 = \langle X_1, X_2 \rangle \quad (9)$$

Относительно мультиинтервалов все рассматриваемые операции с н-переменными являются дистрибутивными. Поскольку разрабатываемый метод предполагается использовать совместно с системой видеораспознавания ориентиров, то ориентир наблюдаем, только когда он

попадает в поле зрения камеры. Следовательно, для каждой области можно определить раствор углов, под которыми может быть наблюдаем каждый ориентир. Вводится ещё одна функция присваивания $w^{<}$, которая пересчитывает н-переменную угла в зависимости от маяка-положения ориентира. Для крайних точек области определяются углы, под которыми виден тот или иной ориентир, образуя максимально возможный интервал. Т.к. может быть видно несколько ориентиров одновременно, то будет получено столько же углов, под которыми эти ориентиры видны, каждому углу соответствует своя область:

$$w^{<}(X, \{O_i\}) = \langle X_j \rangle \tag{10}$$

Также у области, для которой выполняется пересчет углов, имеется свой стартовый угол до пересчета. Чтобы привести все эти области в один мультиинтервал без пересечений, опять используется функция w^{+} (4).

Когда была определена максимально возможная область нахождения робота, приходится переходить к процедуре дискретизации, т.к. н-переменные в данной задаче имеют нелинейную зависимость, и тем самым не позволяют сузить область до приемлемых значений. Для того, чтобы решить эту проблему, полученная область равномерно разбивается на $N \times M$ равных областей, операцией $w^{\#}$. Далее каждая из полученных областей проверяется на соответствие расстоянию до маяка и на соответствие диапазону углов. Области, которые не удовлетворяют данным критериям, исключаются из выборки. Обозначим эту функцию присваивания w^{-}. Области, которые «пережили» отбор, образуют мультиинтервал уточненного положения робота. На этом этапе полученный мультиинтервал и рассчитанное ранее положение робота x'_r (3) обрабатываются функцией проверки коррекции c (1). Соответственно, если рассчитанное принадлежит полученному мультиинтервалу, то оно сохраняется. В противном случае положение робота изменяется на значение, принадлежащее итоговому мультиинтервалу, максимально приближенное по расстоянию и углу к рассчитанному значению x'_r.

$$c(x'_r, X) = \begin{cases} x'_r, \ x'_r \in X \\ x_r, \ x'_r \notin X, \ x_r \in X, \ \left| x_r - x'_r \right| = min \end{cases} \tag{11}$$

где $|x_r - x'_r|$ - расстояние между точками. Полученный мультиинтервал объединяется одной областью операцией w^*, которая содержит в себе все элементы мультиинтервала, эта область будет являться входными данными для следующей итерации алгоритма:

$$X_r^{t_2} = w^* \cdot w^- \cdot w^{\#} \cdot w^{<} \cdot w^{\times} \left(X_o^{t_2}, w^{+}\left(X_r^{t_1} \right) \right) \tag{12}$$

3 Эксперименты

Эксперименты проводились на модели, написанной в среде Python с использованием библиотеки Python Robotics [Sakai, 2018], робот двигался по замкнутой траектории радиусом 10 м в среде, в которой было размещено восемь ориентиров. Гауссовый шум с нулевым средним, помноженный на стандартное отклонение, добавлялся в положение робота и определение расстояния до ориентиров для эмуляции погрешностей датчиков. Стандартное отклонение для эмуляции одометрии составляла 1м для расстояния и 0.5 радиан для угла, для определения расстояния стандартное отклонение составляло 0.2 м. Расстояние до ориентира поступало на вход методов локализации, когда ориентир попадал в поле зрения робота (угол раствора 120 градусов), и когда оно не превышало заданного максимума (10 м). Соответственно данные одометрии и о расстояниях подавались на вход трех методов: расширенный фильтр Калмана, фильтр частиц и предложенный метод на основе н-вычислений.

Рис. 2. Среда моделирования

Эффективность работы методов локализации сравнивалась по отклонению с истинным значением.

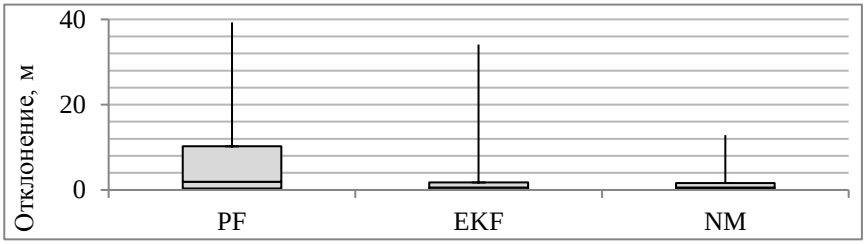


Рис. 3. Диаграмма размаха для сравниваемых методов локализации, PF – фильтр частиц, EKF – расширенный фильтр Калмана, NM – н-вычисления

Результаты отклонений приведены в виде диаграмм размаха, где «усами» показаны максимальное и минимальное значение, средней линией

указана медиана, а высота столбиков соответствует дисперсии распределения. На диаграмме (Рис. 3) видно, что фильтр Калмана и предложенный метод на основе н-вычислений примерно схожи по медиане и дисперсии, однако у фильтра Калмана выше выбросы, которые происходят ввиду нелинейности рассматриваемой задачи. Фильтр частиц показывает хуже результаты в среднем. Но т.к. фильтр частиц опирается на генератор случайных чисел, то в одних случаях он показывает приемлемые результаты, а в других нет, на Рис. 4 представлены данные для «удачного» и «неудачного» случая на фоне средних данных для Н-вычислений.

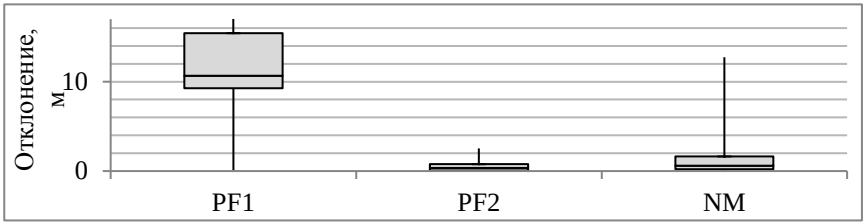


Рис. 4. "Удачный" (PF2) и "неудачный" (PF1) случай для фильтра частиц на фоне средних значений для н-вычислений (NM)

Как видно из графика, фильтр частиц для «удачного» случая превосходит методы другие методы локализации, однако такие результаты при количестве частиц равным 300 были достигнуты примерно в 50% случаев. При увеличении количества частиц в два раза, «неудачные» случаи были сокращены примерно до 35%, однако время работы алгоритма выросло более чем в два раза.

При рассмотрении поведения методов на отрезках с разным количеством видимых ориентиров следует, что для областей, где не видно ориентиров, фильтр Калмана показывает лучшие результаты, т.к. подстраивается под конкретную задачу, в то время как фильтр частиц и метод на Н-вычислениях полагаются в этом случае полностью на одометрию.

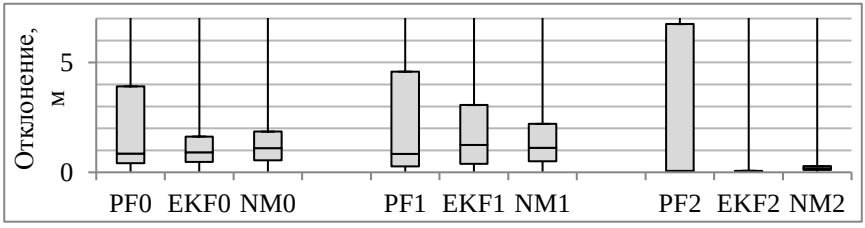


Рис. 5. Диаграмма разброса для областей с 0, 1 и 2 ориентирами

Для случая с одним ориентиром минимальный разброс у предложенного метода, для варианта с двумя ориентирами H-модели показывают результат немного хуже, т.к. применятся механизм дискретизации с шагом 0.5 м.

Поставленные эксперименты в сравнении с фильтром Калмана и фильтром частиц выявили следующие особенности исследуемых методов:

Табл. 1. Сравнительная таблица методов локализации

Метод	**EKF**	**Фильтр частиц**	**H-модели**
Быстродействие	высокое	низкое	среднее
Чувствительность к стартовым усл.	низкое	низкое	низкое
Скорость перестройки при отклонении	высокая	низкая	средняя
Чувствительность к нелинейности	высокое	низкое	низкое
Стабильность	высокая	средняя	высокая
Расширяемость	низкая	высокая	средняя

Заключение

Предложенный метод на основе парадигмы H-вычислений на рассматриваемой задаче локализации по одометрии и видимым ориентирам показал большую стабильность (немного в угоду точности), чем фильтр частиц, и в то же время большую точность и меньшую чувствительность к нелинейности, чем расширенный фильтр Калмана.

Список литературы

[Einicke, White, 1999] Einicke G.A., White L.B. Robust Extended Kalman Filtering // IEEE Trans. Signal Process. 1999. Т. 47. № 9. с. 2596–2599.

[Moral Del, 1996] Moral P. Del. Non Linear Filtering: Interacting Particle Solution // Markov Process. Relat. Fields. 1996. Т. 2. № 4. с. 555–580.

[Sakai, 2018] Sakai A. Python Robotics [Электронный ресурс]. URL: https://github.com/AtsushiSakai/PythonRobotics (дата обращения: 17.02.2019).

[Thrun, Wolfram, Fox, 2005] Thrun S., Wolfram B., Fox D. Probabilistic Robotics (Intelligent Robotics and Autonomous Agents). , 2005.

[Карпов, 2009] Карпов В.Э. О некоторых особенностях применения недоопределенных моделей в робототехнике // Международная научно-практическая конференция «Интегрированные модели и мягкие вычисления в искусственном интеллекте» (28-30 мая 2009) Сб. научных трудов.Т.1. М.: Физматлит. , 2009. с. 520–532.

[Нариньяни, 1986] Нариньяни А.С. Недоопределенность в системах представления и обработки знаний // Известия АН СССР. Техн. кибернетика. 1986. Т. 5.

[Нариньяни, 2007] Нариньяни А.С. Введение в недоопределенность // Информационные технологии. 2007. № 4. с. 1–32.

УДК 681.518

ПОЛУЧЕНИЕ ДАННЫХ ДЛЯ ФОРМИРОВАНИЯ ТРЕХМЕРНОЙ ДОРОЖНОЙ СЦЕНЫ ПО РАДИОЛОКАЦИОННЫМ ДАННЫМ

А.В. Аверин (*averin-artem2007@yandex.ru*)
И.А. Костин (*kostin.ivan.a@gmail.com*)
Н.В. Панокин (*nik_p@mail.ru*)

Национальный исследовательский технологический университет «МИСиС», Москва

Аннотация. Данная статья посвящена проблемам проектирования систем технического зрения на базе технологии миллиметровой радиолокации. Рассмотрена архитектура программного обеспечения для моделирования работы миллиметрового радиолокатора с использованием математических моделей, имитирующих различные объекты дорожной сцены. Проведено исследование трехмерных дорожных сцен различных конфигураций с учетом влияния подстилающей поверхности. Показана возможность классификации дорожных препятствий и подвижных агентов дорожной сцены путем анализа радиолокационных данных с автомобильного радара.

Ключевые слова: трехмерная дорожная сцена, радиолокатор, классификация объектов.

Введение

Формирование реалистичной трехмерной модели дорожной сцены необходимо для отработки методов обработки и интерпретации трехмерных данных с целью повышения эффективности детектирования и распознавания объектов. Для этого необходимо обеспечить решение следующих задач [Якимов и др., 2015]: синтеза проезжей части и стационарных объектов; синтеза реалистичной динамики виртуального мира в условиях ограниченности вычислительных ресурсов.

Данные задачи могут быть решены, в свою очередь, для моделей разного уровня детализации, которые принято делить на:

- *макроскопические* – модели, позволяющие анализировать дорожные объекты с точки зрения взаимодействия потока частиц (пример пакет TRANSIMS (TRansportation ANalysis and SIMulation System) [Angshuman G], [Патент, RU2601133C2], [Barcelo, 2005]);

- *микроскопические* – модели, позволяющие моделировать действия небольшого числа объектов дорожной сцены с помощью правил, определяющих геометрические, скоростные параметры объекта, а также различные маневры движения и взаимодействия объектов между собой (пример пакет VISSIM [Патент, RU2601133C2], [Lownes, 2006]);

- *мезоскопические* – модели, которые занимают среднее положение по уровню детализации между макро- и микроскопическими моделями и призваны описывать на высоком уровне детализации объекты и на низком уровне взаимодействие объектов (пример пакет Paramics [Патент, RU2601133C2], [Cameron, 1994]).

Получение данных о трехмерной дорожной сцене возможно с помощью сенсорного подхода и имитационного моделирования. Сенсорный подход основан на анализе данных с камер и лидаров, которые формируют систему технического зрения. Данные, получаемые с этих систем, можно промоделировать в существующих средах имитационного моделирования AirSim, CARLA, V-Rep, Gazebo. Вариант сенсорного подхода получения радиолокационных данных [Беляев, 2018] в среде CARLA (Car Learning to Act) позволяет создавать изображения дорожной сцены, а также семантическую сегментацию и карту глубины (рисунок 1).

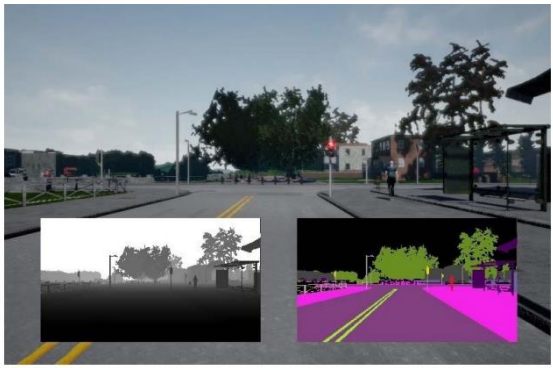


Рис. 1. Среда моделирования CARLA.

Имитационный подход получения радиолокационных данных вплоть до частот 79 ГГц и выше реализован в среде WaveFarer [WaveFarer, Электронный ресурс]. Имитационные средства получения радиолокационной информации часто являются узкоспециализированными и не всегда позволяют в полной мере исследовать качество методов детектирования и распознавания объектов трехмерной дорожной сцены в реальных условиях быстроменяющейся дорожной обстановки. На рынке отсутствуют среды моделирования радиолокационных систем ближнего

(Short Range Radar - SRR) и дальнего (Long Range Radar – LRR) действий [Беляев, 2018], которые сочетали бы возможность имитационного и сенсорного подходов получения радиолокационных данных с возможностью отработки методов детектирования и распознавания трехмерных объектов.

Таким образом, существует необходимость разработки методов математического, в том числе, имитационного моделирования с целью выяснения эффективности разрабатываемых методов обработки радиолокационной информации и уточнения требуемых характеристик 3D радара, используемого для построения трехмерной дорожной сцены.

2 Архитектура программного обеспечения для формирования трехмерной дорожной сцены

Пример построения математической и имитационной модели радиолокатора, на которой основана разрабатываемая модель для системы формирования трехмерной дорожной сцены, описана в [Отчет, RFMEFI57815X0130]. Функционал описанной модели был расширен, в частности, внесением следующих доработок: введено частотное качание луча в плоскости угла места, введена возможность моделирования радара в диапазонах 24 и 77 ГГц, введена возможность моделирования объектов дорожной сцены, введен учет амплитудно-частотных рассогласований в приемных каналах, разработан модуль визуализации результатов обработки радиолокационного сигнала, введен учет влияния подстилающей поверхности.

В радаре используется микрополосковая антенная решетка с частотным качанием луча в плоскости угла места и последующим цифровым диаграммоформированием в горизонтальной плоскости. Характеристики данного радара приведены в таблице 1.

Проектирование программного обеспечения для моделирования системы обработки радиолокационной информации проводилось в соответствии с разработанной архитектурой. Архитектура программного обеспечения (ПО) для обработки радиолокационной информации в случае сигнала с линейной частотной модуляцией и формирования трехмерной дорожной сцены разработана согласно основным принципам проектирования [Гамма, 2001] и отражает основные требования верхнего уровня ко всем составляющим частям системы обработки.

Таблица. 1. Основные характеристики 3D-радара

Параметр	Значение
Центральная частота	24,5 ГГц
Ширина полосы частот	1 ГГц
Количество каналов на передачу	1
Количество каналов на прием	8
Количество элементов в патче	16
Угол обзора по азимуту	101 град
Угол обзора по углу места	16 град
Максимальная дальность действия	67 м

Общая структура ПО представлена на рисунке 2 состоит из: модуля дорожной сцены; модуля формирования сигналов биений; диспетчера (синхронизатора); модуля автокалибровки; модуля первичной обработки; модуля вторичной обработки; модуля визуализации.

Как показано на рисунке 2, архитектура системы предполагает обработку данных, полученных из различных источников. Сигналы биений могут быть получены непосредственно от радара и обработаны сразу, либо сначала записаны на информационный носитель системы хранения данных (например, жесткий диск), а затем прочитаны с него для обработки, либо получены от модуля формирования сигналов биений. В первых двух случаях, при этом, будут обрабатываться реальные данные, а в последнем – смоделированные.

Основной задачей модуля имитации дорожной сцены является вычисление текущих координат объектов дорожной сцены в соответствии с их законами движения. Поскольку каждый объект в рассматриваемой модели трехмерной дорожной сцены является распределенным объектом, его можно представить в виде системы из N отражателей [Бондарев, 1990].

Модуль формирования сигналов биений позволяет создавать имитационные сигналы, обработка которых позволяет оценивать следующие характеристики: максимальная и минимальная дальность при работе 3D радара заданной конфигурации; разрешающая способность по углу, скорости, дальности; минимальное отношение сигнал/шум, обеспечивающее эффективную работу методов обработки сигнала; степень соответствия построенной трехмерной дорожной сцены реальной дорожной сцене.

В модуле первичной обработки радиолокационной информации проводится поэтапное применения БПФ сначала по времени, потом по пространству.

На вход модуля вторичной обработки радиолокационной информации поступает вещественный массив данных, содержащий следующую информацию: количество задетектированных объектов в кадре, координаты каждого из объектов, спектральная плотность мощности.

Модуль автоматической компенсации амплитудно-фазовых рассогласований в приемных каналах АР необходим для учета меняющихся во времени внутренних флуктуаций приемо-передающего тракта радара, которые приводят к амплитудно-фазовым искажениям (рисунок 3). Результатом амплитудно-фазовых искажений являются: искажение формы главного лепестка диаграммы направленности и увеличение уровня боковых лепестков. В разработанной модели предусмотрено несколько типов распределенных объектов, наполняющих трехмерную дорожную сцену: одиночная блестящая точка, человек, легковой автомобиль, грузовой автомобиль, мотоцикл, столбы по краям дороги. Параметры остальных объектов данного типа задаются индивидуально, путем определения параметров каждой блестящей точки, входящей в состав моделируемого распределенного объекта.

Кроме того, модуль имитации дорожной сцены предусматривает расширение списка моделируемых объектов путем добавления новых систем блестящих точек и написания программных функций, описывающих законы движения новых объектов.

3 Классификация объектов трехмерной дорожной сцены

Помимо эвристических методов, основанных на определении меры расстояния между отметками (аналогично методу имитации отжига [Кирсанов, 2007]), применяется широко распространенный метод k-средних [Coates, 2012], суть которого заключается в минимизации квадратичного отклонения отметок от центров соответствующих им объектов.

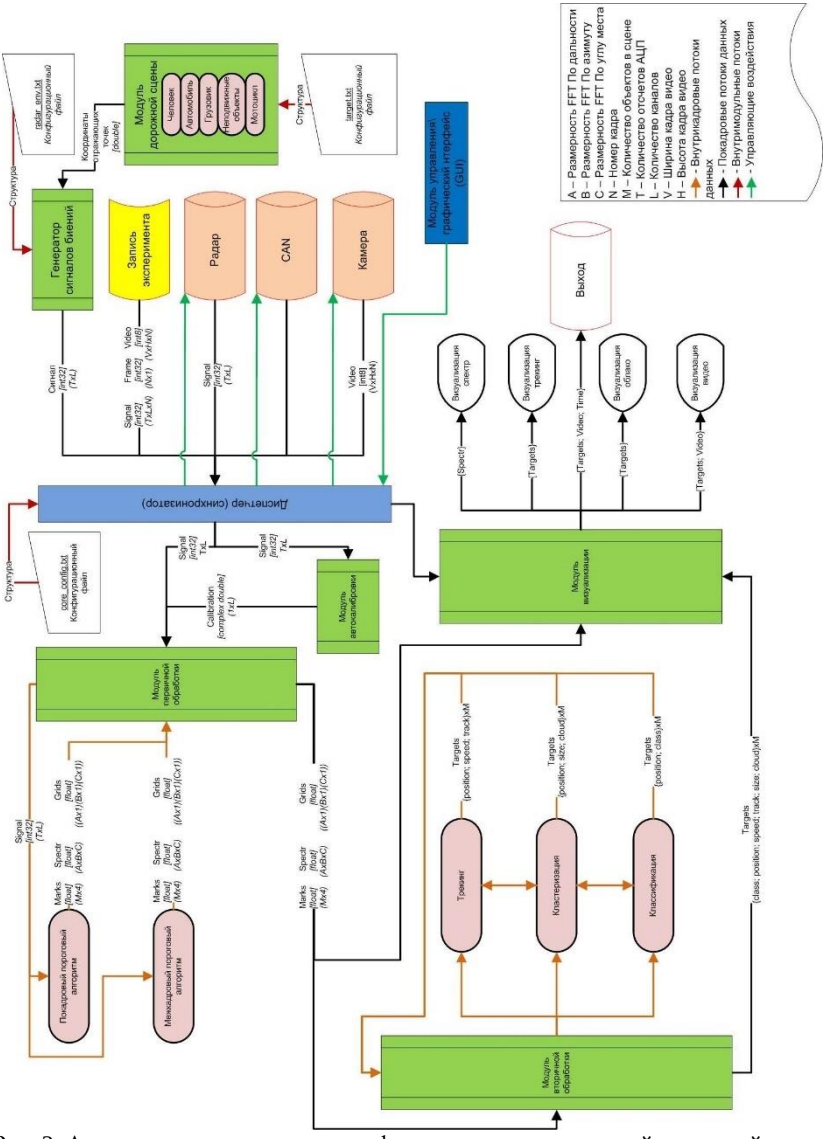
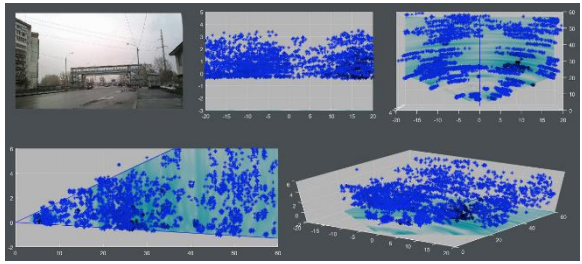
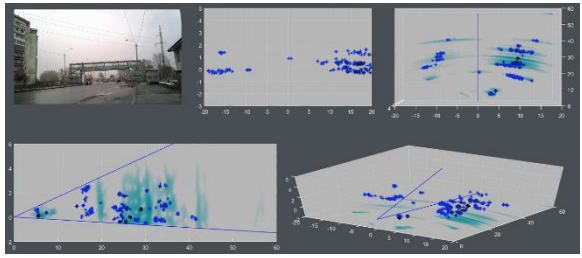


Рис. 2. Архитектура приложения для формирования трехмерной дорожной сцены

Квалификация может проводиться как для отдельных отметок, так и для выделенных кластеров. Целью квалификации является определение типа объекта, выделение препятствий, дорожных конструкций, других транспортных средств и иных участников дорожного движения. В классической теории радиолокации и распознавании образов квалификация целей базируется на априорной информации о количестве и качественном составе классов объектов, причем известно, что с ростом числа классов ее достоверность падает [Горелик, 1963]. Набором признаков в данном случае является вектор $x_e = [R, V, \theta, \varphi, l, a]$ в $\mathbf{R}^6$, где R – радиальная дальность, V – радиальная скорость, θ – направление по азимуту, φ – направление по углу места, l – оценка линейного размера, a – ЭПР.



(а)



(б)

Рис. 3. Результат первичной обработки радиолокационной информации до автоматической компенсации амплитудно-частотных рассогласований каналов АР (а) и после нее (б)

Однако, в настоящее время широко распространены технологии машинного и глубокого обучения для обнаружения и квалификации объектов трехмерной дорожной сцены, где в качестве исходных данных выступают характеристики массивов отражающих точек (рисунок 4, 5).

Одним из таких методов автоматического анализа трехмерной дорожной сцены по радиолокационному изображению является подход, основанный на сверточных нейронных сетях [Huijing Zhao, 2012; Wang, 2003] для предоставления информации о количестве и расположении значимых объектов на трехмерной дорожной сцене (рисунок 6).

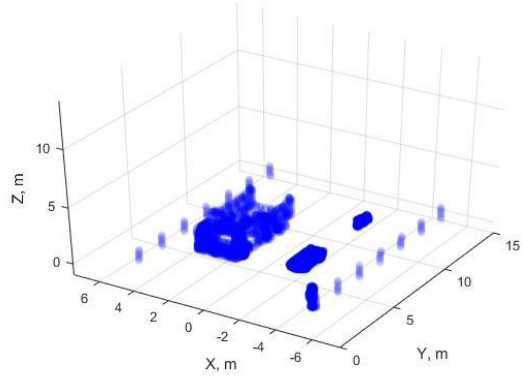


Рис.4. Пример моделируемой трехмерной дорожной сцены (отбойники по краям дороги, человек у края проезжей части, грузовой автомобиль, легковой автомобиль, мотоцикл)

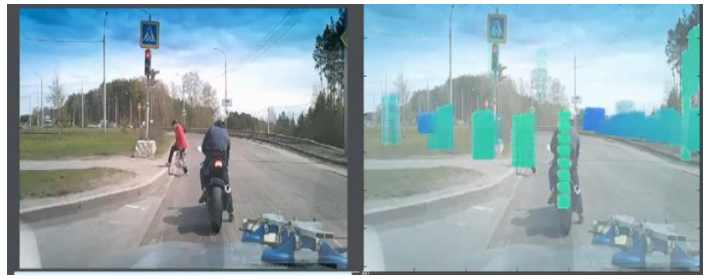


Рис.5. Кластеры отражающих точек с наложением на видеоизображение (цвет отображает расстояние до объекта)

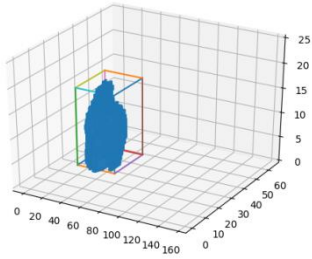


Рис.6. Локализация объекта известного класса при глубоком обучении

Реальные данные были получены с помощью разработанного прототипа радара, работающего в диапазоне 24ГГц (рисунок 7).

(а) (б)

Рис. 7. 3D-радар 24ГГц, (а) – на штативе, (б) – установлен на автомобиль

Список литературы

[Беляев, 2018] А.А. Беляев, Т.А. Суанов, Д.О. Троц. Моделирование работы автомобильного радара в задаче автономного движения. Инженерный вестник Дона, №2 (2018).

[Бондарев, 1990] Бондарев Л.А. Отражающие свойства моделей сложных радиолокационных целей. – Радиотехника. – 1990. – № 7. С. 8-13.

[Гамма, 2001] Э. Гамма, Р. Хелм, Р. Джонсон, Дж. Влиссидес. Приемы объектно-ориентированного проектирования. Паттерны проектирования. – СПб: Питер, 2001. – 368с.: ил. (Серия «Библиотека программиста»).

[Горелик, 1963] А.Л. Горелик, Ю.Л. Барабаш. Селекция и распознавание на основе локационной информации. – М.: Радио и связь, 1990 – 240 с. || Ю.Л. Барабаш, Б.В. Варский, В.Т. Зиновьев. Автоматическое распознавание образов – Киев: КВАИУ, 1963 – 168 с.

[Кирсанов, 2007] Кирсанов М.Н. Графы в Maple. – М.: Физматлит, 2007. – С.151-154

[Патент, RU2601133C2] «Имитационная модель движения транспортных и пешеходных потоков в городских условиях на основе агентно-ориентированного подхода (RU2601133C2)».

[Отчет, RFMEFI57815X0130] Отчет о прикладных научных исследованиях и экспериментальных разработках. «Разработка грузового автомобиля повышенной безопасности, оснащённого программным комплексом на базе технологий миллиметровой радиолокации для целей обнаружения и классификации препятствий и других транспортных средств с обеспечением функции управления торможением» по соглашению с МОН № 14.578.21.0130 от 27.10.2015, уникальный идентификатор проекта RFMEFI57815X0130.

[Якимов и др., 2015] П.Ю. Якимов, С.А. Разлацкий. Применение метода Хафа для детектирования объектов в трехмерной сцене. Информационные технологии и нанотехнологии (ИТНТ-2015): материалы Международной конференции и молодежной школы. – Самара: Изд-во СамНЦ РАН, 2015. – с. 324-328.

[Angshuman G] Angshuman G., Introduction to TRANSIMS.

[Barcelo, 2005] Barcelo J. Microscopic traffic simulation: a tool for the design, analysis and evaluation of intelligent transport systems // Journal of intelligent and robotic systems. - 2005. - Vol. 41, №2-3. - P. 173-203.

[Cameron, 1994] Cameron G. PARAMICS - moving vehicles on the connection machine / G. Cameron, B. Wyline, D. McArthur // Proceedings of the conference on supercomputing. - IEEE computer society press, 1994. - P. 291-300.

[Coates, 2012] Adam Coates and Andrew Y. Ng. Learning Feature Representations with K-means, Stanford University, 2012

[Lownes, 2006] Lownes N.E. VISSIM: a multi-parameter sensitivity analysis / N.E. Lownes, R.B. Machemehl // Proceedings of the 38th conference on Winter simulation. - 2006. - P. 1406-1413.

[Wang, 2003] C.C. Wang, C. Thorpe, and S. Thrun, “Online Simultaneous Localization and Mapping with Detection and Tracking of Moving Objects: Theory and Results from a Ground Vehicle in Crowded Urban Areas,” in Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2003.

[Huijing Zhao, 2012] Huijing Zhao, Jie Sha, Yipu Zhao, Junqiang Xi, Jinshi Cui, Hongbin Zha and Ryosuke Shibasaki. Detection and Tracking of Moving Objects at Intersections Using a Network of Laser Scanners. Article (PDF Available) in IEEE Transactions on Intelligent Transportation Systems 13(2):655-670, 2012

[WaveFarer, Электронный ресурс] https://www.remcom.com/wavefarer-automotive-radar-software. (дата обращения 3.12.2015)

УДК 004.932.7, 004.4, 004.021

ВОССТАНОВЛЕНИЕ КАРТ ГЛУБИН ИЗОБРАЖЕНИЙ, ПОЛУЧЕННЫХ С ЕДИНСТВЕННОЙ ВИДЕОКАМЕРЫ В РЕАЛЬНОМ ВРЕМЕНИ НА ПЛАТФОРМЕ NVIDIA JETSON TX2

К.Ф. Муравьев (*muravev.kf@phystech.edu*)
Московский физико-технический институт
Федеральный исследовательский центр "Информатика и управление" Российской академии наук, Москва

А.В. Боковой (*bokovoy@isa.ru*)
Федеральный исследовательский центр "Информатика и управление" Российской академии наук, Москва
Российский университет дружбы народов, Москва

Аннотация. В статье рассматривается практическое применение искусственных нейронных сетей для восстановления карт глубин изображений, полученных с единственной камеры малой робототехнической системы, в контексте задачи одновременного картирования и локализации по видеопотоку (vision-based Simultaneous Localization and Mapping – vSLAM) в режиме реального времени. Нейронные сети обучены на актуальных коллекциях данных и протестированы на встраиваемом компьютере NVidia Jetson TX2, который благодаря низкому энергопотреблению, малым размерам и особенностям архитектуры позволяет ускорить параллельные вычисления на борту робототехнической системы. Приводятся результаты экспериментов с разными архитектурами нейронных сетей, а также дается описание программных оптимизаций, позволяющих добиться работы алгоритмов восстановления глубины изображений в реальном времени.[1]

Ключевые слова: восстановление глубины, vSLAM, нейронные сети, Nvidia Jetson.

Введение

В последнее время мобильные роботы и беспилотные летательные аппараты все чаще используются в различных коммерческих и бытовых

1 Работа выполнена при финансовой поддержке РНФ (проект 16-11-00048)

целях. Для навигации малых мобильных роботов широко применяются методы одновременного картирования и локализации по видеопотоку (vision-based SLAM, vSLAM) [Blösch M. et al, 2010][Fraundorfer F. et al., 2012][Yang S. et al, 2016]. Методы vSLAM получили широкое распространение, так как они не требуют наличия на борту робототехнического устройства никаких датчиков, кроме единственной видеокамеры, и делают возможной навигацию в помещениях, где использование спутниковой навигации затруднено. Классические методы vSLAM, основанные на извлечении структуры из движения (Structure from Motion, SfM), имеют существенные недостатки, такие, как потеря структуры при поворотах робота на месте и невозможность восстановления точного масштаба карты (все расстояния на построенной карте - относительные) [Davison A. J. et al., 2007][Klein G. et al., 2007]. Восстановление карты глубин по видеопотоку позволяет устранить данные недостатки, сведя задачу vSLAM к задаче одновременного картирования и локализации с использованием видеокамеры и датчиков глубины, для которой разработаны эффективные методы решения [Enders F. Et al., 2012][Kerl C. et al., 2013].

Классические методы восстановления глубины по видеоданным основаны на использовании оптического потока [Newcombe R. A. et al, 2010]. Как правило, производительность таких методов недостаточна для обработки видеопотока в реальном времени с помощью бортовых вычислителей робототехнических систем. В настоящее время помимо классических алгоритмов восстановления глубины применяются также нейросетевые методы, обрабатывающие отдельно каждый кадр видеопоследовательности [Laina I. et al, 2016][Garg R. et al., 2016][Kuznietsov V. et al., 2017]. Подобные методы позволяют достичь приемлемого качества восстановления глубины и производительности, достаточной для обработки видеопотока в реальном времени, но требуют наличия мощного графического ускорителя, что затрудняет их применение для навигации беспилотных транспортных средств малого размера. В данной работе рассматривается применение нейросетей для восстановления глубины на одноплатном компьютере NVIDIA Jetson TX2, который оснащен графическим ускорителем и при этом достаточно компактен и энергоэффективен для использования на борту малых робототехнических систем. Описываются программные оптимизации, позволяющие сократить время обработки одного изображения на Jetson TX2 до 80 мс в разрешении 320x240 и 150 мс в разрешении 640x480.

1 Описание нейросети

Для восстановления глубины используются нейросети с полносверточной архитектурой, не содержащие полносвязных слоев (fully-convolutional networks [Long J. et al., 2015][Dai J. et al., 2016]). Предсказание карты глубины сверточным слоем вместо полносвязного позволяет значительно уменьшить число параметров нейросети и, как следствие, сократить объем занимаемой памяти и ускорить процесс предсказания карты глубины.

Используемые в работе нейросети принимают на вход цветное трехканальное изображение и выдают предсказанную карту глубины. Они состоят из двух частей: свертки и развертки. Сверточная часть (энкодер) содержит серию слоев свертки (convolution) и субдискретизации (pooling) и последовательно уменьшает размерность изображения, преобразуя его в набор высокоуровневых признаков. В данной работе в качестве сверточной части используется архитектура ResNet-50 [He K. et al., 2016], предобученная на коллекции изображений ImageNet. Разверточная часть (декодер) представляет собой серию операций транспонированной свертки (Deconvolution, [Dumoluin V. et al., 2016][Zeiler M. D. et al., 2010]) и активаций. Декодер преобразует набор высокоуровневых признаков, предсказанных энкодером, в карту глубины. Карта глубины представляет собой двумерный массив той же ширины и высоты, что и входное изображение, в каждой ячейке которого находится глубина соответствующего пикселя входного изображения. Глубина задается положительным числом с плавающей точкой.

Для экспериментов на NVIDIA Jetson использовались две нейросетевые архитектуры. Первая нейросеть обрабатывает изображения размером 640x480 пикселей. Ее архитектура представлена на рисунке 1. Энкодером является предобученная сеть ResNet-50 без полносвязных слоев. Энкодер преобразует входное изображение в карту признаков размерности 20x15x2048. Разверточная часть начинается со свертки с ядром размера 1x1 и 1024 фильтрами. Затем следуют 5 блоков развертки, в которых последовательно применяются операции нормализации, транспонированной свертки (Deconvolution) с шагом 2 и ядрами размера 5x5, активации (ReLU). Количество фильтров в Deconvolution-слоях блоков развертки уменьшается последовательно с 512 до 32. После блоков развертки следует сверточный слой с одним фильтром, выводящий предсказанную карту глубины. Вторая нейросеть обрабатывает изображения размером 320x240 пикселей. Энкодером является предобученная сеть ResNet-50 без полносвязных слоев и последнего блока сверток. Энкодер преобразует входное изображение в карту признаков размерности 20x15x1024. Разверточная часть содержит свертку с ядром

размера 1x1 и 1024 фильтрами и 4 блока развертки, идентичные разверточным блокам первой нейросети. После блоков развертки следует сверточный слой с одним фильтром, выводящий предсказанную карту глубины.

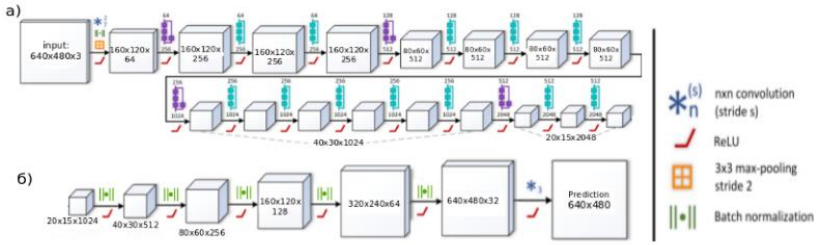


Рисунок 1. Архитектура нейросети для разрешения 640x480. а) архитектура энкодера; б) архитектура декодера

Обучение нейросетей производилось на коллекции NYU Depth Dataset, содержащей около 1100 видеосцен и более 400 тысяч кадров, снятых в помещениях различного типа, с размеченными с помощью сенсора LIDAR картами глубины. Глубина изображений ограничивалась 10 метрами. Функцией потерь являлась среднеквадратичная ошибка между истинной и предсказанной глубиной. Валидация проводилась на двух выборках: первая содержала кадры из сцен, отсутствующих в обучающей выборке, но снятых в помещениях схожего с обучающей выборкой типа, а вторая — кадры из сцен, снятых в помещениях типа NYU office (офисы Нью-Йоркского университета). Тип помещений NYU Office в обучающей выборке отсутствовал.

Обучение проводилось в четыре этапа:

1. Обучаются только веса декодера, веса энкодера не меняются
2. Обучаются веса декодера и последнего ResNet-блока
3. Обучаются веса декодера и двух последних ResNet-блоков
4. Обучаются все веса нейросети

Такой подход позволяет ускорить процесс адаптации нейросети с предобученными слоями к новой задаче и повысить качество работы [Yosinsky J. et al., 2014].

По итогам обучения, среднеквадратичное отклонение (RMSE) предсказаний на изображениях размера 640x480 составило 0.78м на первой валидационной выборке и 0.88м на второй, на изображениях размера 320x240 — 0.87м на первой валидационной выборке и 0.92м на второй. Графики обучения представлены на рисунке 2.

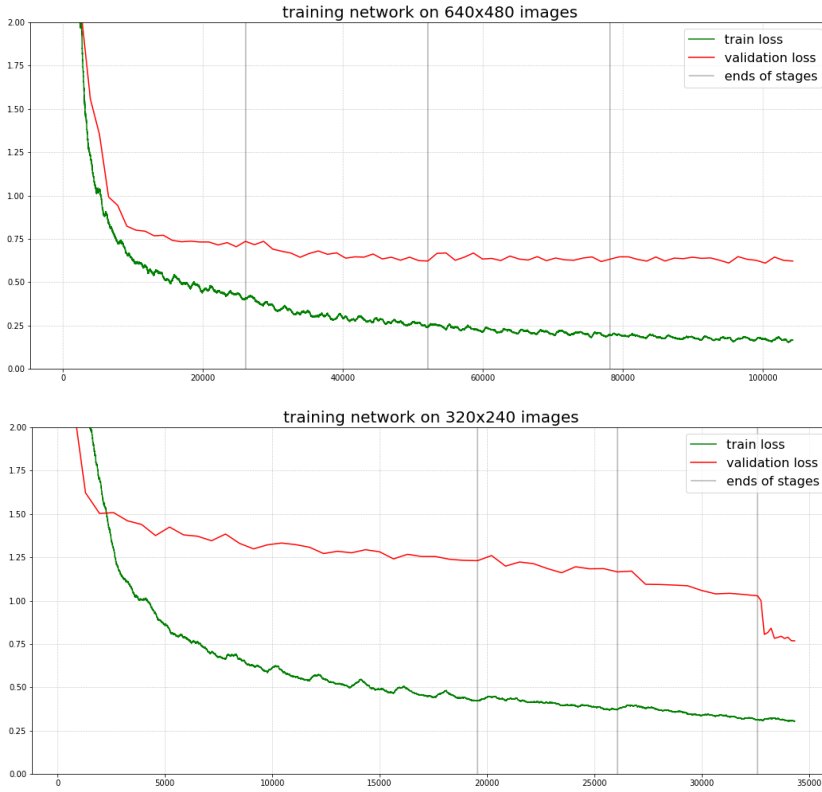


Рисунок 2. Графики обучения нейросетей для восстановления глубины. Сверху график обучения на изображениях разрешения 640x480, снизу — 320x240. Вертикальными линиями показаны границы этапов обучения. По оси абсцисс — число шагов обучения, по оси ординат — средний квадрат ошибки (MSE). Для валидации использовалась первая валидационная выборка

Ошибка на валидационной выборке намного превосходит ошибку на обучающей выборке почти на всем протяжении обучения. Такое различие объясняется отсутствием в валидационной выборке кадров из помещений, присутствующих в обучающей выборке и большими различиями в интерьере и в условиях съемки разных помещений.

2 Особенности реализации

Архитектура нейросети была построена с использованием библиотеки глубинного обучения TensorFlow и ее расширения Keras. Код, реализующий архитектуру, был написан на языке Python. Для борьбы с переобучением после каждого блока развертки был включен слой дропаута (Dropout) с коэффициентом 0.5.

Обучение нейросетей проводилось на гибридном высокопроизводительном вычислительном кластере ФИЦ ИУ РАН [ГВВК, 2018]. Так как обучающая коллекция данных полностью не помещалась в оперативную память, то для ускорения процесса обучения она была разбита на 10 частей, которые последовательно загружались в оперативную память и разбивались на батчи для подачи на вход нейросети.

Тестирование алгоритмов восстановления глубины проводилось на встраиваемом компьютере NVIDIA Jetson TX2 с операционной системой Ubuntu 16.04 и пакетом инструментов JetPack 3.0 [Buonaiuto N. et al., 2017][Hadidi R. et al., 2018]. Данный компьютер оснащен 256-ядерной видеокартой с архитектурой PASCAL и 6-ядерным центральным процессором CPU Complex ARMv8 и имеет 8 ГБ оперативной памяти, разделяемой между GPU и CPU. При этом он имеет размеры 50x87 мм, а его энергопотребление составляет 10-13 Вт на максимальной тактовой частоте, что позволяет встраивать его в малогабаритные робототехнические системы, в том числе в малые БПЛА. Высокая скорость работы нейронных сетей на NVIDIA Jetson достигается за счет поддержки библиотек CuDNN и TensorRT, а также аппаратной поддержки вычислений с половинной точностью (fp16).

Для эффективной работы нейросети на встраиваемом компьютере она была переведена в формат TensorRT engine с поддержкой вычислений с половинной точностью. Код, осуществляющий обработку полученных с камеры изображений и визуализацию предсказанной карты глубины в реальном времени, был реализован на языке C++ с использованием технологии CUDA. Изображение с камеры захватывалось с помощью библиотеки Gstreamer и записывались в видеопамять с помощью CUDA. Затем изображение переводилось в формат RGBA и нормализовалось для подачи на вход нейросети. Конвертация в RGBA и нормализация были распараллелены с помощью CUDA. Далее нормализованное изображение подавалось на вход нейросети для восстановления карты глубины. Восстановленная нейросетью карта глубины и исходное изображение отрисовывались на экране с помощью библиотеки OpenGL.

Исходный код восстановления глубины по изображениям в реальном времени доступен в Github-репозитории[2]. Нейросети в формате TensorRT engine доступны в облачном хранилище Яндекс.Диск[3].

3 Результаты экспериментов

Эксперимент проводился на платформе NVIDIA Jetson TX2. Изображения, полученные с установленной на платформе видеокамеры, подавались на вход нейросети. Вычисленная нейросетью карта глубины вместе с исходным изображением визуализировались с помощью библиотеки OpenGL. Примеры визуализации представлены на рисунке 3.

Рисунок 3. Визуализация восстановления глубины в реальном времени на NVIDIA Jetson по изображениям в разрешении 640x480. а) Трехканальное изображение с камеры, установленной на платформе; б) Предсказанная карта глубины в формате grayscale; в) Предсказанная карта глубины, визуализированная в цветовой гамме

Средняя частота восстановления глубины при разрешении 320x240 составила 13 Гц. Средняя частота восстановления глубины при разрешении 640x480 составила 6 Гц.

Для оценки производительности библиотеки TensorRT на NVIDIA Jetson было проведено сравнение работы нейросети для восстановления глубины в различных фреймворках на различных устройствах по скорости и объему занимаемой памяти. Результаты сравнения представлены в таблице 1. Для сравнения с NVIDIA Jetson TX2 использовался ноутбук, оснащенный 8-ядерным процессором Intel Core i7-8550 и видеокартой NVIDIA GeForce MX150 (384 CUDA-ядра, архитектура Pascal), имеющей энергопотребление 30 Вт на максимальной тактовой частоте.

В ходе сравнения выяснилось, что перевод нейросети из формата Keras model в формат TensorRT engine позволил ускорить ее работу более чем в 4 раза, а также сократить в 3 раза объем потребляемой оперативной памяти и в 5-8 раз — объем потребляемой памяти GPU. Также выяснилось, что NVIDIA Jetson по производительности в задаче восстановления глубины

2 https://github.com/CnnDepth/jetson-inference/tree/master/fcrn-camera
3 https://yadi.sk/d/fgSHHUpgw_aw4w

почти не уступает ноутбуку с видеокартой MX150, имеющей вдвое большую тепловую мощность, чем видеокарта NVIDIA Tegra X2, установленная на Jetson.

Таблица 1. Сравнение производительности нейросети в различных фреймворках на различных устройствах

Фреймворк	Устройство	Разрешение	Время обработки одного кадра, мс	Объем занимаемой оперативной памяти, ГБ	Объем занимаемой памяти GPU, ГБ
TensorRT	Jetson TX2	640x480	152	0.62	0.31
Keras	Jetson TX2	640x480	630	1.9	2.65
TensorRT	Ноутбук	640x480	143	0.77	0.51
TensorRT	Jetson TX2	320x240	80	0.62	0.28
Keras	Jetson TX2	320x240	340	1.8	2.6
TensorRT	Ноутбук	320x240	73	0.77	0.37

Выводы

В результате работы были созданы и обучены нейронные сети для восстановления карты глубины по изображениям с единственной видеокамеры в реальном времени. Качество восстановления глубины было протестировано на коллекции NYU Depth Dataset, содержащей изображения из помещений различного типа с глубинами до 10 метров. Среднеквадратичная ошибка на тестовой выборке данной коллекции составила 0.87м при разрешении 640x480 и 0.92м при разрешении 320x240. Для достижения высокой скорости работы нейросетей на встраиваемом компьютере они были переведены в формат TensorRT engine с поддержкой вычислений половинной точности. При тестировании на платформе NVIDIA Jetson TX2 частота построения карты глубины составила 13 Гц при разрешении 320x240 и 6 Гц при разрешении 640x480. Такая скорость работы в совокупности с приемлемым качеством восстановления глубины делает возможным применение данных нейросетей в методах одновременного картирования и локализации, в том числе для навигации малых беспилотных летательных аппаратов. В дальнейшем планируется использование восстановленных с помощью подобных нейросетей карт глубины в методах SLAM, основанных на данных с единственной видеокамеры, с помощью фреймворка ROS [ROS, 2009] и алгоритма RTABMap [Labble M. Et al. 2011].

Список литературы

[Blösch M. et al. 2010] Blösch, M., Weiss, S., Scaramuzza, D., & Siegwart, R. Vision based MAV navigation in unknown and unstructured environments //Robotics and automation (ICRA), 2010 IEEE international conference on. – IEEE, 2010. – С. 21-28

[Fraundorfer F. et al., 2012] Fraundorfer, F., Heng, L., Honegger, D., Lee, G. H., Meier, L., Tanskanen, P., & Pollefeys, M. Vision-based autonomous mapping and exploration using a quadrotor MAV //2012 IEEE/RSJ International Conference on Intelligent Robots and Systems. – IEEE, 2012. – С. 4557-4564.

[Yang S. et al, 2016] Yang S., Scherer S. A., Zell A. Robust Onboard Visual SLAM for Autonomous MAVs //Intelligent Autonomous Systems 13. – Springer International Publishing, 2016. – С. 361-373

[Laina I. et al., 2016] Laina I, Rupprecht C., Belagiannis V., Tombari F., Navab N. Deeper depth prediction with fully convolutional residual networks //3D Vision (3DV), 2016 Fourth International Conference on. – IEEE, 2016. – С. 239-248.

[Garg R. et al., 2016] Garg R., Kumar V., Carneiro G., Reid I. Unsupervised cnn for single view depth estimation: Geometry to the rescue //European Conference on Computer Vision. – Springer, Cham, 2016. – С. 740-756.

[Kuznietsov Y. et al., 2017] Kuznietsov Y., Stückler J., Leibe B. Semi-supervised deep learning for monocular depth map prediction //Proc. of the IEEE Conference on Computer Vision and Pattern Recognition. – 2017. – С. 6647-6655.

[Davison A. J. et al., 2007] Davison A. J. et al. MonoSLAM: Real-time single camera SLAM //IEEE Transactions on Pattern Analysis & Machine Intelligence. – 2007. – №. 6. – С. 1052-1067.

[Klein G. et al., 2007] Klein G., Murray D. Parallel tracking and mapping for small AR workspaces //Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality. – IEEE Computer Society, 2007. – С. 1-10.

[Enders F. et al, 2012] Endres F. et al. An evaluation of the RGB-D SLAM system //Icra. – 2012. – С. 1691-1696.

[Kerl C. et al, 2013] Kerl C., Sturm J., Cremers D. Dense visual SLAM for RGB-D cameras //2013 IEEE/RSJ International Conference on Intelligent Robots and Systems. – IEEE, 2013. – С. 2100-2106.

[Long J. et al, 2015] Long J., Shelhamer E., Darrell T. Fully convolutional networks for semantic segmentation //Proceedings of the IEEE conference on computer vision and pattern recognition. – 2015. – С. 3431-3440.

[Dai J. et al, 2016] Dai J. et al. R-fcn: Object detection via region-based fully convolutional networks //Advances in neural information processing systems. – 2016. – С. 379-387.

[Newcombe R. A. et al., 2010] Newcombe R. A., Davison A. J. Live dense reconstruction with a single moving camera //2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. – IEEE, 2010. – С. 1498-1505.

[He K. et al., 2016] He K. et al. Deep residual learning for image recognition //Proceedings of the IEEE conference on computer vision and pattern recognition. – 2016. – С. 770-778.

[Dumoluin V. et al., 2016] Dumoulin V., Visin F. A guide to convolution arithmetic for deep learning //arXiv preprint arXiv:1603.07285. – 2016.

[Zeiler M. D. et al, 2010] Zeiler M. D. et al. Deconvolutional networks. – 2010.

[Yosinsky J. et al., 2014] Yosinski J. et al. How transferable are features in deep neural networks? //Advances in neural information processing systems. – 2014. – С. 3320-3328.

[Buonaiuto J. et al., 2017] Buonaiuto N. et al. Satellite identification Imaging for small satellites using NVIDIA. – 2017.

[Hadidi R. et al., 2018] Hadidi R. et al. Real-time image recognition using collaborative IoT devices //Proceedings of the 1st on Reproducible Quality-Efficient Systems Tournament on Co-designing Pareto-efficient Deep Learning. – ACM, 2018. – С. 4.

[ГВВК, 2018] Федеральный исследовательский центр Информатика и управление РАН [Электронный ресурс]: сайт. – Москва: ФИЦ ИУ РАН. – URL: http://hhpcc.frccsc.ru (дата обращения: 12.09.2018)

[ROS, 2009] Quigley M. et al. ROS: an open-source Robot Operating System //ICRA workshop on open source software. – 2009. – Т. 3. – №. 3.2. – С. 5.

[Labble M. Et al. 2011] Labbé M., Michaud F. Memory management for real-time appearance-based loop closure detection //2011 IEEE/RSJ International Conference on Intelligent Robots and Systems. – IEEE, 2011. – С. 1271-1276.

УДК 621.865.8-5

ЛОКАЛИЗАЦИЯ И НАВИГАЦИЯ МУЛЬТИАГЕНТНОЙ РОБОТОТЕХНИЧЕСКОЙ СИСТЕМЫ НА ОСНОВЕ ARUCO-МАРКЕРОВ

И. М. Толстой (*tolstoy.i.m@yandex.ru*)
К. С. Захаров (*kon7666007@yandex.ru*)
Санкт-Петербургский институт информатики и автоматизации Российской академии наук, Санкт-Петербург

И. А. Кан (*igorkaan@yandex.ru*)
Санкт-Петербургский государственный университет аэрокосмического приборостроения, Санкт-Петербург

Аннотация. Авторами работы рассмотрена проблема перемещения мобильных роботов в закрытом помещении. Представлена реализация системы навигации мультиагентной робототехнической системы на основе ArUco-маркеров с применением классических алгоритмов планирования пути. Описан метод оценки положения и ориентации роботов по изображению, получаемому с камеры, расположенной под потолком помещения, а также процесс передачи данных между управляющим сервером и агентами по Wi-Fi. Представлена модель симуляции перемещения роботов в среде V-REP. Рассмотрены такие вопросы, как планирование пути, локализация и навигация мобильных роботов.

Ключевые слова: мультиагентные робототехнические системы, локализация, навигация, планирование пути, мультиагентные системы, автономные роботы, ArUco-маркеры.

Введение

В настоящее время мультиагентные робототехнические системы (МАРС) являются одним из наиболее перспективных направлений в робототехнике. Возможные области их применения варьируются от оборонной сферы до индустрии развлечений [Манько и др., 2015]. В зависимости от задач, поставленных перед МАРС, выбирается одна из трех стратегий группового управления: централизованная, децентрализованная, иерархическая [Лохин и др., 2014]. Авторами данной статьи реализована централизованная стратегия управления мобильными роботами в замкнутом пространстве на основе информации, получаемой с

монокулярной камеры, а также разработана виртуальная модель для тестирования алгоритмов построения пути и поведения агентов.

1 Конструкция роботов-агентов

Единичный модульный робот представляет из себя мобильную робототехническую платформу, состоящую из основного корпуса, управляющей электроники, элемента питания, сменных навесных модулей и мотор-редукторов. Размеры платформы: длина – 210 мм, ширина – 170 мм, высота – 90 мм, диаметр колес – 60 мм. Крепление модулей осуществляется посредством соединения «Ласточкин хвост» с последующей фиксацией винтом. Также на корпусе расположен ArUco маркер, по которому сервер распознает робота. Основным элементом управления платформой является микроконтроллер ESP32 с 32-битным микропроцессором, OnChip BlueTooth и Wi-Fi модулем. Алгоритмы соединения, движения и обработки команд от сервера зашиты в микроконтроллер и реализованы на языке программирования C++.

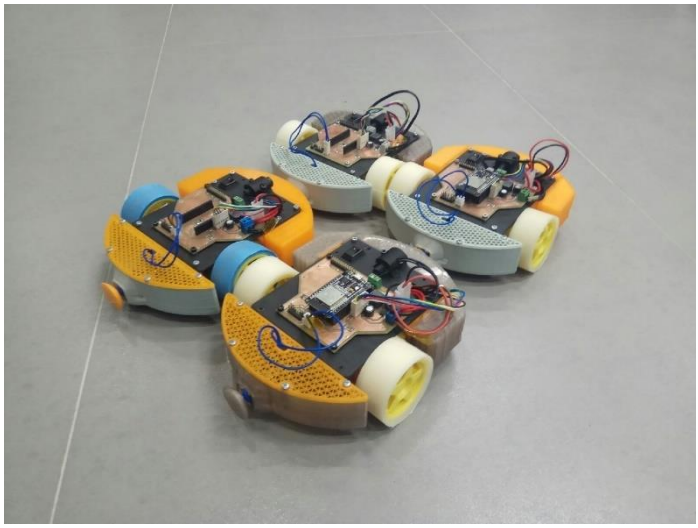


Рис. 1. Внешний вид роботов-агентов.

2 Локализация роботов-агентов

2.1 Выбор метода локализации

Одним из ключевых аспектов управления роботами-агентами является получение информации о местоположении и ориентации каждого робота в

пространстве. Существуют различные методы позиционирования мобильных роботов. В работе [Aswin N Raghavan et al., 2010] описан подход, использующий технологию Bluetooth. При проведении экспериментов авторы использовали 3 Bluetooth модуля, образовывающих треугольник со сторонами 5,28 м, 2,62 м, 3,27 м в помещении 6 м на 8 м. Робот выполнял серию случайных перемещений с последующими поворотами, после которых с интервалом в 3 с производилось 5 запросов к модулям Bluetooth для определения уровня принимаемого сигнала с последующим усреднением по пяти полученным значениям. Затем авторами был применен метод итеративной трилатерации для определения положения робота. Среднее значение ошибки позиционирования робота в составило 0,427 ± 0,229 м. Таким образом, данный подход не обеспечивает достаточную точность локализации, что неприемлемо для функционирования мультиагентной системы мобильных роботов, так как среднее значение ошибки в обоих случаях превышает размер робота. Помимо этого существуют так называемые системы глобального позиционирования, например, GPS, однако, они не работают внутри закрытых помещений [Aitor De Blas et al., 2017]. Так же для локализации роботов внутри помещений применяются RFID-метки, однако их использование требует предварительной подготовки среды и кропотливой калибровки, прежде чем система позиционирования начнет работать корректно [Martinelli, 2015]. Кроме вышеперечисленных способов, существует метод локализации на основе маркеров дополненной реальности (ArUco, ArTag, AprilTag и др.), позволяющих определить позицию маркера и его угол поворота относительно плоскости изображения, что дает информацию об ориентации робота. Данный подход был успешно применен в работе [Pickem et al., 2017]. Семейство меток ArUco было выбрано исходя из наличия в библиотеке OpenCV [OpenCV, 2019] модуля для работы с ним. Опытным путем авторами данной статьи было определено среднее значение погрешности позиционирования маркера равное 0,25 м. При этом расстояние от камеры до маркера составляло 3 м, а размер маркера – 5 см. Такое значение погрешности не превышает длину робота, а значит является удовлетворительным. На основании вышеизложенного для определения положения робота в пространстве был выбран последний подход.

2.2 Получение данных о местоположении роботов-агентов

Для получения данных о местоположении и ориентации роботов использовался модуль ArUco библиотеки OpenCV, предоставляющий алгоритмы обнаружения маркера дополненной реальности на изображении и для вычисления его пространственного местоположения. Таким образом, имея на роботе соответствующую маркировку, можно определить его

положение и ориентацию в системе координат изображения, получаемого с камеры. Пример подобной локализации робота представлен на рис. 2.

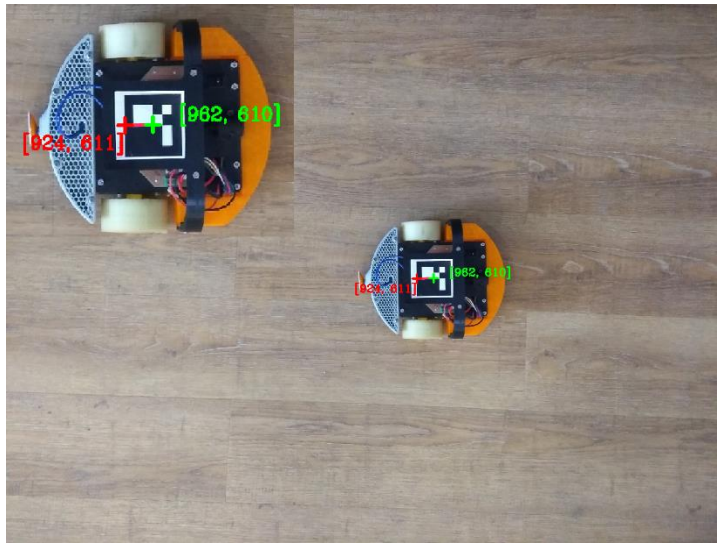


Рис. 2. Результат локализации робота агента.

На рис. 2 представлено изображение, полученное с камеры. Зеленым крестом отмечен центр маркера робота, а красным – конец вектора, проведенного из центра и описывающего его ориентацию. Координаты вектора в пространстве изображения, также представленные на рисунке, имеют значения (962, 610) и (924, 611) для начала и конца вектора соответственно.

3 Управление роботами

Передача управляющих сигналов агентам осуществляется по протоколу MQTT. На рис. 3 представлена блок-схема управления роботом при движении по построенному маршруту.

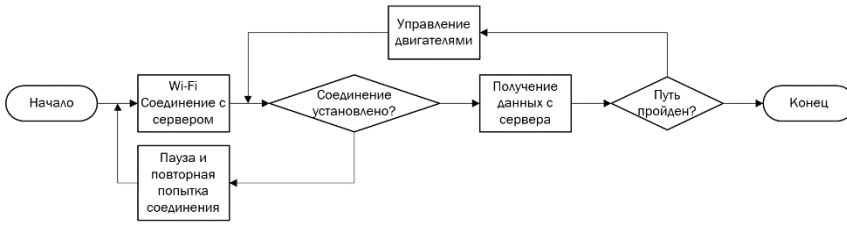


Рис. 3. Алгоритм управления роботом при движении по построенному маршруту.

Как видно из рис. 3, платформа, подключенная к серверу, получает данные по идентификатору вида “platform/N”, где N – уникальный id, соответствующий id маркера, закрепленного на роботе. Таким образом, робот получает команду на поворот вокруг своей оси, остановку или движение прямо, а также величину угла отклонения от заданной точки маршрута, необходимую для работы ПИД-регуляторов, после чего, на основании этих данных, происходит управление двигателями.

4 Алгоритм планирования пути

В работе [Захаров и др., 2018] был проведен анализ алгоритмов планирования пути, и в качестве наиболее подходящего для задачи реконфигурации группы роботов в двумерном пространстве был выбран алгоритм RRT-Connect [JJ Kuffner Jr et al., 2000]. Данный алгоритм строит два дерева − из начальной и конечной точки − и заканчивает работу тогда, когда они соединятся в одной точке. Деревом называются множество точек в пространстве, расположенных последовательно и соединённых между собой отрезками. Поиск пути для одного робота с помощью алгоритма RRT-Connect занимает меньше секунды на компьютере с процессором Intel Core i5 3230M.

Алгоритм планирования принимает следующие входные данные: начальная точка, конечная точка, шаг планировщика (максимальное расстояние между точками пути) и данные о препятствиях. Эти данные можно получить как из компьютерной симуляции, так и из реального эксперимента, следовательно, этот алгоритм планирования пути подойдёт как для реального робота, так и для компьютерной модели.

Для предотвращения коллизий алгоритмом движения робота предусмотрена его остановка при приближении на определённое расстояние к другому роботу. Из двух роботов продолжит движение тот, длина оставшегося пути которого больше, а второй будет ждать, пока первый не отъедет на безопасное расстояние, и уже после продолжит движение по маршруту. Если же остановка одного из двух роботов не предотвращает столкновения, происходит перестроение маршрута второго робота, в объезд первого.

5 Проведение экспериментов

5.1 Среда моделирования

Для проведения компьютерных экспериментов был выбран бесплатный симулятор роботов V-REP, так как в нём реализованы возможности,

необходимые в контексте задачи планирования пути для группы роботов: каждым объектом на сцене можно управлять с помощью встроенного скрипта, ROS или BlueZero узла или удалённого API (Application Programming Interface), имеется возможность фиксирования столкновений, а также 4 физических движка, благодаря которым результаты компьютерного эксперимента могут быть приближены к результатам реального эксперимента. Контроллеры могут быть написаны на языках C/C++, Python, Java, Lua, Matlab или Octave. Для реализации алгоритма локализации и навигации роботов с V-REP использовался язык Python.

5.2 Процесс симуляции

Для тестирования работы алгоритма и движения роботов по заданному пути была использована модель Pioneer p3dx [Generationrobots, 2019]. На первом шаге работы программы происходит локализация всех объектов на сцене и отправка данных планировщику. На втором шаге происходит нахождение путей для каждого робота. На третьем шаге, после того как все пути сгенерированы, каждый робот начинает движение по собственному маршруту, что проиллюстрировано на рис. 4.

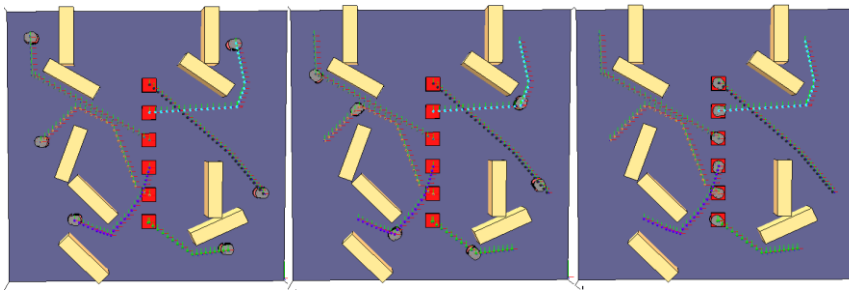


Рис. 4. Процесс симуляции построения пути и движения роботов по маршруту.

На рис. 4 представлен процесс симуляции построения пути и движения группы роботов. Серыми точками представлены роботы, маршрут каждого из которых показан в виде последовательности точек определённого цвета. Препятствия, в обход которых строится маршрут, представлены как параллелепипеды желтого цвета, а целевыми точками для каждого робота являются центры красных квадратов. На левой части рис. 4 роботы находятся в исходных позициях, на средней – роботы в процессе движения, а на левой – результат передвижения роботов по построенным маршрутам. Как видно из рис. 4, каждый робот занял необходимую позицию на поле.

5.3 Эксперименты на реальной платформе

При проведении экспериментов использовалась камера Logitech C525, размещенная на высоте 2 м и охватывающая квадрат 1,5 м на 1,5 м. Для построения маршрута движения, помимо самих роботов, маркерами были отмечены целевые точки, к которой платформы должны прийти, и препятствие, которое платформы должны объезжать. На рис. 5 демонстрируется процесс передвижения трех роботов по построенным траекториям.

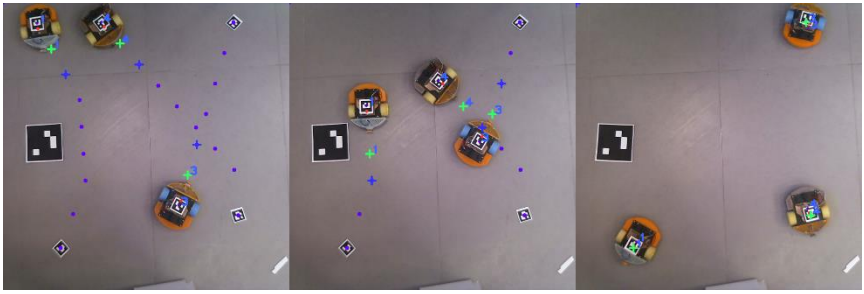


Рис. 5. Результат построения маршрута для реальной платформы.

Фиолетовыми кругами отмечены точки траектории движения в объезд препятствий. Зелеными крестами отмечены текущие точки маршрута, а синими – следующие за ними. Как видно из рис. 5, построенные траектория движения для робота 1 успешно огибает препятствие. В ходе проведения испытаний роботы успешно достигали конечных точек маршрута, однако были выявлены проблемы перерегулирования поворота платформы по углу и скорости движения, связанные с малым диапазоном допустимых значений управляющего воздействия на двигатели и малой чувствительностью к сигналам управления.

5.4 Анализ точности передвижения платформы.

В ходе испытаний было проведено 50 запусков реальной платформы как по автоматически сгенерированному пути, так и пути, заданному вручную. На рис. 6 представлен один из графиков прохождения роботом заданного пути.

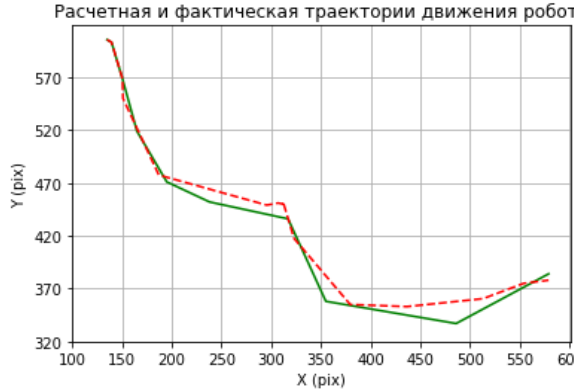


Рис. 6. График прохождения роботом пути.

На приведенном выше графике непрерывной зеленой линией отмечена расчетная траектория движения робота, а пунктирной красной – фактическая. Относительно точек расчетной траектории среднее отклонение фактического положения платформы составило 12 пикселей изображения, что составляет 0,5 от длины платформы. Для V-REP симуляции среднее значение отклонения робота от траектории составило 0,1 от размера робота. Для всей серии экспериментов среднее значение отклонения положения робота от траектории движения составило 0,63 от длины робота.

6 Заключение

В результате данного исследования был предложен и практически реализован метод централизованного управления мультиагентной робототехнической системой на основе применения ArUco-маркеров и алгоритма построения пути RRT-Connect. Симуляция работы алгоритма планирования маршрута и движения роботов показала пригодность данного подхода при управлении мобильными роботами. Эксперименты, проведенные с реальными платформами, также подтвердили эффективность предложенных методов, несмотря на проблемы перерегулирования угла поворота и скорости движения роботов. В будущем планируется модернизация роботов с целью обеспечения более точного управления двигателями для устранения вышеописанных недостатков, а также учет реального размера робота для более корректного построения траектории движения.

Список литературы

[Захаров и др., 2018] Захаров К.С., Ватаманюк И.В., Крестовников К.Д. Алгоритмы самореконфигурации робототехнических систем. // Робототехника и техническая кибернетика. №4(21). СПб: ЦНИИ РТК, 2018.

[Лохин и др., 2014] Лохин В.М., Манько С.В., Романов М.П., Диане С.А-К. Перспективы применения, принципы построения и проблемы разработки мультиагентных робототехнических систем. // XII всероссийское совещание по проблемам управления ВСПУ-2014. Москва, 2014

[Манько и др., 2015] Манько С.В., Лохин В.М., Романов М.П. Концепция построения мультиагентных робототехнических систем. // Вестник МГТУ МИРЭА. №3. Москва, 2015

[Aitor De Blas et al., 2017] A. Blas, D. López-de-Ipiña. Improving trilateration for indoors localization using BLE beacons. // 2nd International Multidisciplinary Conference on Computer and Energy Science (SpliTech). 2017.

[Aswin N Raghavan et al., 2010] Aswin N Raghavan, Harini Ananthapadmanaban, Manimaran S Sivamurugan, Balaraman Ravindran. Accurate mobile robot localization in indoor environments using Bluetooth. // IEEE International Conference on Robotics and Automation. 2010.

[Generationrobots, 2019] Pioneer P3-DX mobile robot. - https://www.generationrobots.com/en/402395-robot-mobile-pioneer-3-dx.html

[JJ Kuffner Jr et al., 2000] JJ Kuffner Jr, LaValle S.M. RRT-Connect: An Efficient Approach to Single-Query Path Planning. // IEEE International Conference on Robotics and Automation. 2000.

[Martinelli, 2015] F. Martinelli. A robot localization system combining RSSI and Phase Shift in UHF-RFID signals. // IEEE Transactions on Control Systems Technology. 2015. Vol. 23. p. 1782 – 1796.

[OpenCV, 2019] OpenCV Library. - https://docs.opencv.org

[Pickem et al., 2017] D. Pickem, P. Glotfelter, L. Wang, M. Mote, A. Ames, E. Feron, M. Egerstedt. The Robotarium: A remotely accessible swarm robotics research testbed. // IEEE International Conference on Robotics and Automation (ICRA). 2017.

УДК 681.51

ЗАДАЧА ЛОКАЛИЗАЦИИ БЕСПИЛОТНОГО ТРАНСПОРТНОГО СРЕДСТВА С ИСПОЛЬЗОВАНИЕМ ДСМ-МЕТОДА

Д.А. Добрынин (minirobot@yandex.ru)
ФИЦ ИУ РАН, Москва,
РГГУ, Москва

Аннотация. В статье описывается задача определения локального положения мобильного робота с использованием ДСМ-метода. Для получения входной информации могут использоваться различные датчики расстояния, видеокамеры и стереокамеры. Эта входная информация преобразуется в набор признаков, который характеризует положение робота в окружающей его среде. ДСМ-система используется как классификатор, позволяющий по набору признаков определить положение робота в пространстве с точностью до некоторой области. Приведены качественные результаты процесса обучения и параметры локализации.

Ключевые слова: ДСМ-метод, машинное обучение, визуальная локализация.

Введение

Локализация, т.е. определение положения и ориентации мобильного робота является необходимой и критически важной функцией системы управления роботом для правильного функционирования внутри помещения. Из-за электромагнитной экранировки внутри железобетонных зданий не работают приемники систем GPS/ГЛОНАСС и магнитные компасы. Поэтому для определения местоположения робота в помещениях приходится привязываться к визуальным характеристикам объектов помещения. Для этого могут использоваться лазерные дальномеры и видеокамеры. Использование инерциальных датчиков возможно только на небольших промежутках времени, поскольку высокий уровень шума приводит к быстрой потере точности измерения скорости и расстояния. Аналогичная проблема наблюдается при использовании датчиков длины пути. Проскальзывание колес, неровности, деформация шин приводит к быстрому нарастанию погрешности измерения пути, и как следствие к потере точности позиционирования.

Для визуальной локализации используется большое количество подходов. Технологии SLAM используют различные подходы с использованием лазерных дальномеров, видеокамер, стерео и 3D камер. Широкое распространение получили методы сегментации изображений с дальнейшей обработкой с помощью нейронных сетей. Например, в работе [Буйвал, 2017] используется алгоритм визуальной локализации с использованием граней изображений как основного характера визуального признака.

Одними из перспективных являются методы, основанные на обучении, позволяющие получить требуемый результат и обойтись без построения сложных математических моделей [Добрынин, 2017].

1. Постановка задачи

В данной работе рассматривается следующий вариант задачи локализации робота в помещении, представленный на рис. 1:

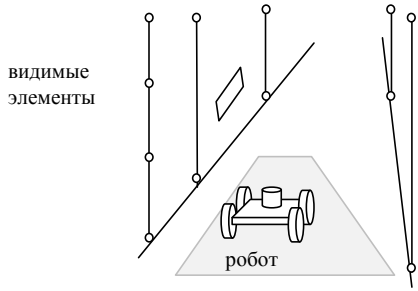


Рис. 1 Видимые элементы сцены.

Робот имеет на борту систему наблюдения, с помощью которой он пытается локализовать свое положение в пространстве. В качестве такой системы наблюдения могут использоваться различные одометрические датчики – ИК датчики расстояния, ультразвуковые датчики, лазерный дальномер, или видеокамера, 3D камера и т.п. Напрямую использовать информацию с датчиков затруднительно, поскольку она представлена в «сыром» виде и объем ее довольно велик, особенно для 3D систем. Перед подачей на вход классификатора информацию о расстояниях или изображение с камеры нужно преобразовать в элементарные примитивы. Отметим что, несмотря на большие различия в типах используемых датчиков, процессы поиска примитивов похожи друг на друга. Рассмотрим, какие данные можно получить для разных датчиков.

При использовании ИК и ультразвуковых датчиков на выходе получаем расстояние для фиксированных направлений по отношению к курсу робота. Полученные данные необходимо преобразовать в диапазоны значений, которые будут определять необходимые зоны. Выделить из такого небольшого набора данных еще что-нибудь не представляется возможным.

Лазерный дальномер является более сложной одометрической системой и выдает расстояния сразу для целой сетки направлений. Из такой информации можно выделить не только диапазоны значений, но и получить более подробную информацию об окружающих объектах, например – выделить последовательности точек, образующие прямые, проверить – образуют ли прямые угол и т.п.

Видеокамера дает кадр, обработкой которого можно выделить вертикальные, горизонтальные, наклонные прямые, углы, замкнутые контуры и более сложные объекты. Можно выделять не только геометрическую форму объектов, но и использовать информацию о его цвете.

Использование 3D камер или 3D лазерных дальномеров помимо прямых и контуров позволяет выделять плоскости и различные поверхности. Алгоритмы обработки в этом случае существенно сложнее.

После процедуры выделения примитивов из исходной информации системы наблюдения видимая сцена будет описываться набором примитивов – диапазонов значений, линий, углов, контуров, плоскостей, п а также их взаимным положением и т.п. Отметим, что выбор примитивов и их параметров сильно влияет на дальнейшее качество распознавания. Поэтому выбор и оптимизация параметров примитивов является нетривиальной задачей.

Методы получения примитивов из входной информации довольно хорошо разработаны. Например, для поиска прямых используется преобразование Хафа. Конкретные методы получения примитивов в данной работе не рассматриваются.

Для разных положений робота можно получить сцены с различным набором примитивов. Задача локализации, т.е. определения положения робота по видимой сцене, сводится к поиску в описании неизвестной сцены заранее полученных наборов примитивов, которые соответствуют сценам с известными координатами. Для этого можно использовать классификаторы, построенные на нейронных сетях, деревьях решений или с использованием других методов ИИ. Мы будем использовать классификатор, построенный на основе ДСМ-системы.

ДСМ классификатор выдает информацию о принадлежности сцены к ранее известным. Таким образом, локализация производится с точностью до некоторой области. Для получения точных координат робота внутри

области необходимо использовать информацию с одометрических датчиков, что является несравненно более простой задачей.

2. ДСМ-метод

Для создания системы управления, способной обучаться, можно построить классификатор входных сигналов с помощью ДСМ метода. Аббревиатура **ДСМ** расшифровывается как Джон Стюарт Милль. Метод назван в честь британского философа XIX века, идеи которого положены в основу метода. ДСМ-метод автоматического порождения гипотез [Финн, 1991] является теорией автоматизированных рассуждений и способом представления знаний для решения задач прогнозирования в условиях неполноты информации. Классический ДСМ метод работает с замкнутым множеством исходных примеров, которое формируется экспертом и составляет базу знаний. Каждый пример описывается множеством элементарных признаков и наличием (или отсутствием) целевого свойства. С помощью специальных логических процедур из этой базы знаний ДСМ-система получает гипотезы, которые объясняют свойства исходных примеров из-за наличия или, наоборот, отсутствия в структуре примеров определенной совокупности признаков. Таким образом, ДСМ система выделяет из исходной информации в базе знаний существенные совокупности признаков, т. е. осуществляет автоматическую классификацию. ДСМ метод успешно применим в тех областях знаний, где пример можно представить в виде множества (или кортежа) элементарных признаков.

В отличие от классического ДСМ метода, который работает с замкнутым множеством исходных примеров и заранее определенными их свойствами, динамический ДСМ метод позволяет работать в открытой среде с неизвестным заранее количеством примеров [Добрынин, 2006].

Динамический ДСМ система работает в двух режимах:

- режим обучения, когда происходит заполнение базы фактов (множество обучающих примеров) и генерируются гипотезы, составляющие базу знаний;
- рабочий режим, когда полученные ранее гипотезы используются для определения принадлежности к области локализаии.

Множество обучающих примеров – это множество пар вида

$$E=\{e_i\}=\{(\mathbf{X}_i,u^i)\},$$

где $\mathbf{X}_i$ – множество примитивов, u^i – указывает на область локализации. Множество примитивов удобно представлять битовой строкой, где один бит соответствует конкретному примитиву. Для кодирования области локализации можно использовать его номер в списке областей локализации.

Операция пересечения (нахождения общей части) двух объектов при использовании битовых строк реализуется с помощью логической функции «побитовое И». Операция вложения, отвечающая на вопрос – входят ли все компоненты объекта А в объект В, реализуется как «побитовое И» элементов объектов А и В, а затем сравнение результата с элементами вкладываемого объекта А.

Гипотезы представляются в виде множества пар вида:

$$G=\{g_i\}=\{\{x_i,y^i\}\},$$

где x_i – часть множества примитивов, y^i – соответствующая область локализации . Гипотезы существуют двух видов: положительные гипотезы определяют, при каком входном множестве примитивов будет достигнута заданная область локализации; отрицательные гипотезы определяют, какие примитивы не должны входить в сцену для данной области локализации.

В режиме обучения для формирования обучающих примеров используется внешняя система определения координат – так называемый «учитель». Совокупность примитивов и соответствующая им область локализации определяет один обучающий пример. Этот пример проверяется на уникальность и заносится ДСМ системой в базу фактов. После занесения каждого нового примера во множество обучающих примеров производится поиск гипотез. На полученные гипотезы могут накладываться дополнительные ограничения, например, запрет на контрпримеры, когда положительная гипотеза не должна вкладываться в отрицательные примеры и наоборот. Эти ограничения определяются используемым ДСМ-методом.

Полученное множество гипотез будет содержать все возможные пересечения (общие части) обучающих примеров. Далее среди них отбираются минимальные гипотезы, то есть такие, которые вкладываются в остальные. Тем самым количество «полезных» гипотез резко сокращается. Полученные минимальные гипотезы проверяются на уникальность и заносятся в базу знаний.

Обучение должно производиться до тех пор, пока база знаний не перестанет пополняться новыми гипотезами. Очевидно, что в этом случае обучающий алгоритм перебрал все возможные варианты входных воздействий, на которые он способен реагировать, и можно считать, что база фактов достаточно полна.

В рабочем режиме ДСМ система получает на вход множество примитивов, найденных для текущей сцены, из которых формируется тестовый вектор. Принятие решения происходит путем проверки вложения гипотез в этот вектор. Если в тестовый вектор для текущей сцены вкладывается гипотеза, то робот находится в области локализации данной гипотезы. Если же ни одной гипотезы не найдено, то это неизвестное положение.

3. Обучение системы локализации

Для обучения ДСМ-системы необходима внешняя система локализации (учитель), которая определяет положение робота в пространстве и выдает область локализации как показано на рис. 2.

В качестве такого учителя может выступать как сам человек, так и любая другая управляющая система.

Основным требованием к обучающей системе управления является непротиворечивость команд управления. В противном случае, такие противоречивые команды могут «ввести в заблуждение» обучаемую систему, что приводит к игнорированию противоречивых входных сигналов, и соответственно, снижению качества обучения.

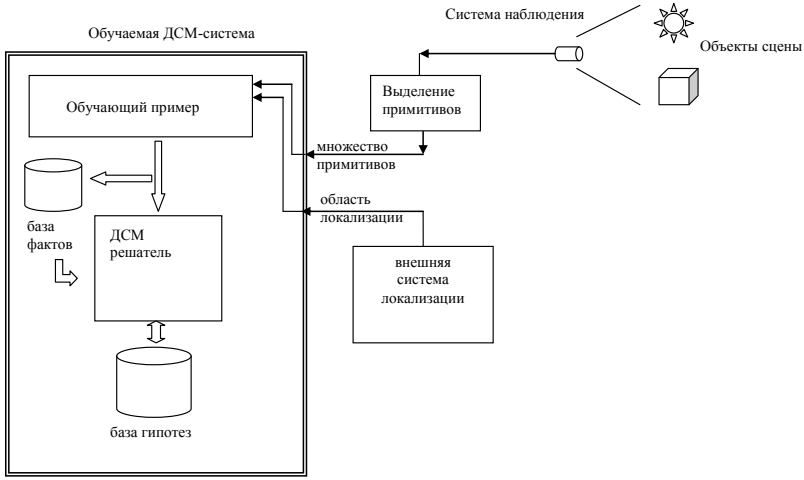


Рис. 2 Обучение ДСМ-системы

Информация с системы наблюдения (датчиков расстояния, видеокамеры и т.п.) предварительно обрабатывается, чтобы получить для текущей сцены множество примитивов, которые определяют уникальность данной сцены.

Сам процесс обучения ДСМ-системы выглядит следующим образом:

- «учитель» определяет положение робота и решает, к какой области локализации он относится в текущий момент. Информация с системы наблюдения преобразуется во множество примитивов. Вся эта информация подается в ДСМ-систему, при этом формируется т.н. «обучающий пример», который ДСМ-система заносит в базу фактов. Если в этой базе фактов такой пример уже есть, то ничего не происходит.

- если появляется новый обучающий пример, ранее не встречающийся в базе фактов, то в этом случае он передается ДСМ-решателю, который формирует с его помощью новую гипотезу. Если полученная гипотеза удовлетворяет критериям непротиворечивости, то она добавляется в базу гипотез.

- пополнение базы фактов и получение новых гипотез производится до тех пор, пока работает режим обучения.

Критерием завершения обучения может служить тот факт, что перестает пополняться база гипотез. Это означает, что на вход обучаемой системы не поступает новая информация. После окончания режима обучения ДСМ-система имеет набор гипотез, которые в дальнейшем используются для работы обученной системы управления. Работа обученной таким способом системы управления описана выше.

Отметим, что поскольку информация с разных датчиков приводится к одному виду – примитиву, то для ДСМ-системы нет разницы, получен ли примитив от ИК датчика расстояния или путем сложной обработки кадра с видеокамеры. Таким образом, возможно комплексирование информации с различных датчиков, которое достигается как бы естественным путем, без чрезмерного усложнения алгоритмов управления.

Замечательным свойством ДСМ-метода является возможность «объяснения» полученных результатов. Каждая гипотеза о причинах, в нашем случае содержит набор необходимых примитивов, достаточных для описания сцены. Анализ гипотез позволяет определить, какие примитивы и связанные с ними методы обработки входной информации являются достаточными для описания сцены, а какие – избыточными. Это дает возможность целенаправленного улучшения методов обработки видеоинформации.

4. Заключение

Использование ДСМ системы в качестве классификатора для определения области локализации мобильного робота дает определенные преимущества:

Эффективность обучения (скорость). Обучение для нейронной сети – принципиально длительный процесс, который требует сотни тысяч тактов для устойчивого обучения. В этом отношении ДСМ-метод обладает несомненным преимуществом - для обучения достаточно получить несколько разных обучающих примеров (в минимальном случае всего два). Поскольку самыми ресурсоемкими компонентами системы являются алгоритмы предварительной обработки визуальной информации, то сокращение количества примеров ведет к эффективному ускорению всего процесса обучения.

Динамическое обучение. Динамический ДСМ метод позволяет эффективно работать с заранее неизвестным количеством примеров при сравнительно небольших вычислительных затратах.

Требуемые ресурсы. При реализации практических алгоритмов встает проблема ограниченности вычислительных ресурсов автономного робота. Для работы ДСМ метода достаточно небольших вычислительных ресурсов. Основную вычислительную мощность потребляют алгоритмы выделения признаков из изображения сцены.

Оценка качества входных данных. ДСМ метод обладает возможностью «фальсификации» избыточных входных данных. Способность «объяснить» полученный результат, в отличие от нейронных сетей, позволяет целенаправленно улучшать используемый набор примитивов для описания сцены, тем самым увеличивая быстродействие и улучшая процесс распознавания.

Прямая локализация. ДСМ классификатор выдает информацию о принадлежности сцены к ранее известным, т.е. локализация производится с точностью до некоторой области. Это свойство можно использовать для упрощения процедуры принятия решений в контуре управления

Эти особенности, особенно потенциально высокая скорость обучения и нетребовательность к вычислительным ресурсам, позволяют выделить динамический ДСМ-метод как один из основных претендентов для построения обучаемой системы локализации для роботов.

Список литературы

[Буйвал, 2017] Buyval A., Gavrilenkov M., Magid E. A multithreaded algorithm of UAVvisual localization based on a 3D model of environment: implementation with CUDAtechnology and CNN filtering of minor importance objects. // In Proceedings of the2017 International Conference on Artificial Life and Robotics (ICAROB 2017). January 19-22, 2017, Miyazaki, Japan. P.356-359.

[Добрынин, 2006] Добрынин Д.А. Динамический ДСМ-метод в задаче управления интеллектуальным роботом.// Десятая национальная конференция по искусственному интеллекту КИИ-2006, 25-28 сентября 2006 г., Обнинск, Труды конференции, М:Физматлит 2006, т.2.

[Добрынин, 2017] Добрынин Д.А. Задача обучения движению по траектории беспилотного транспортного средства с использованием ДСМ-метода // Четвертый Всероссийский научно-практический семинар «Беспилотные транспортные средства с элементами искусственного интеллекта» (БТС-ИИ-2017, 5-6 октября 2017 г., г. Казань, Республика Татарстан, Россия): тр. семинара. / под ред. Е.А. Магида, В.Е. Павловского, К.С. Яковлева – Казань: Центр инновационных технологий, 2017. – 240 с. С.118-125.

[Финн, 1991] Финн В.К. Правдоподобные рассуждения в интеллектуальных системах типа ДСМ //Итоги науки и техники. Сер. «Информатика». Т. 15. - М.: ВИНИТИ, 1991.

УДК 004.8

LPLIAN: АЛГОРИТМ ПЛАНИРОВАНИЯ ТРАЕКТОРИИ С УЧЕТОМ ГЕОМЕТРИЧЕСКИХ ОГРАНИЧЕНИЙ В ДИНАМИЧЕСКОЙ СРЕДЕ

Н.А. Соболева (*nasoboleva@edu.hse.ru*)
Национальный исследовательский университет «Высшая Школа Экономики», Москва

К.С. Яковлев (*yakovlev@isa.ru*)
Федеральный исследовательский центр «Информатика и управление» Российской академии наук, Москва
Национальный исследовательский университет «Высшая Школа Экономики», Москва

Аннотация. Рассматривается задача планирования траектории, удовлетворяющей ограничениям на угол поворота в динамической среде, которая моделируется графом специального вида. Наивным решением этой задачи является повторный запуск алгоритма после каждого возникающего изменения (добавление, удаление препятствий и т.д.). Очевидно, что при незначительных изменениях среды такой подход характеризуется большим объемом повторяющихся вычислений. Для повышения эффективности перепланирования и сокращения числа повторяющихся вычислений предлагается использовать подход Lifelong Planning. В работе предложена модификация алгоритма планирования LIAN, основанная на этом подходе. Приведены результаты модельных экспериментальных исследований.[1]

Ключевые слова: планирование траектории, эвристический поиск, LIAN, Lifelong Planning.

Введение

Традиционно в искусственном интеллекте и робототехнике задача планирования траектории рассматривается как задача поиска пути на графе, вершинам которого соответствуют допустимые положения агента в пространстве, ребрам – элементарные траектории следования из одного положения в другое. При этом весьма распространено использование графа

[1] Работа выполнена при финансовой поддержке РНФ (проект 16-11-00048)

специального вида – графа регулярной декомпозиции (англ. – grid) [Яковлев, 2009][Yap, 2002], [Яковлев и др., 2013], который является простой и вместе с тем информативной графовой моделью окружающей агента среды. Граф регулярной декомпозиции (ГРД) – взвешенный неориентированный граф, каждый элемент (вершина) которого соответствует некоторой области пространства и может быть либо проходим, либо непроходим для агента. Путь на ГРД это последовательность проходимых секций, где секция – отрезок прямой соединяющей две проходимые вершины.

Для поиска пути на ГРД обычно применяются алгоритмы эвристического поиска семейства A* [Hart et al., 1968], такие как Theta* [Daniel et al., 2010], JPS [Harabor, et al., 2011] и другие. При этом подобные алгоритмы обычно минимизируют длину траектории и не накладывают никаких ограничений на ее форму, что может рассматриваться как недостаток при использовании алгоритмов в составе интеллектуальных систем управления мобильными роботами. Так, траектории содержащие резкие смены направления движения (повороты) не всегда являются исполнимыми (т.е. следование по ним не всегда возможно). Одним из способов решения этой проблемы является наложение геометрических ограничений. Методы решения задачи построения траектории с наложенными ограничениями были предложены ранее, в частности алгоритм, учитывающий ограничения на угол поворота, – LIAN [Андрейчук и др., 2017], [Yakovlev et al.,2015], однако эти методы предназначены для планирования в статической среде. Целью данной работы является разработка модификации алгоритма LIAN для функционирования в динамической среде.

1 Постановка задачи

Рассмотрим точечного агента на плоскости, осуществляющего перемещение внутри прямоугольной рабочей области, состоящей из свободного пространства и препятствий: $U = W_{free} \cup W_{obs}$ (здесь $W_{obs} = U \setminus W_{free}$). Дискретная модель этой области представляется в виде ГРД, который в свою очередь может быть представлен в виде матрицы $A_{m \times n}$. (i, j) – элемент матрицы соответствует квадратной области $u_{ij} \in U$, которая центрирована в (i, j), этот элемент помечен как заблокированный, если $u_{ij} \cap W_{obs} \neq \emptyset$ и является проходимым в противном случае. Задана также функция видимости (проходимости), которая определяет, какие перемещения разрешены на ГРД: $los: U \times U \rightarrow \{true, false\}$. Эта функция возвращает true, если отрезок линии, определенный двумя вершинами ГРД a, b – не пересекает ни одну из заблокированных клеток. Путь от начальной

до целевой клетки, от s до g, представляет собой последовательность проходимых сегментов:

$\pi\ (s, g) = \langle (s = a^0, a^1), (a^1, a^2), \dots, (a^k, a^{k+1} = g) \rangle$, где $a^i \in A_{m \times n}$ и $los(a^{i-1}, a^i) = true\ \forall i = 1, \dots, k+1$. Длина пути – сумма длин отрезков пути, $e_i = (a^{i-1}, a^i)$, а длина отрезка равна евклидову расстоянию между его конечными вершинами.

Теперь рассмотрим путь π, составленный из сегментов e_i, и значение $\alpha_m(\pi) = \max(|\alpha(e_1, e_2)|, |\alpha(e_2, e_3)|, \dots, |\alpha(e_{k-1}, e_k)|)$, где $\alpha(e_i, e_{i+1})$ – угол между двумя последовательными отрезками пути. Задача планирования в статической среде может быть сформулирована следующим образом. По данной начальной и целевой клеткам, а также ограничению $\alpha_{MAX} \in [0°, 180°]$ – найти путь π, такой что $\alpha_m(\pi) \leq \alpha_{MAX}$ (см. Рис. 1). Критерием оптимальности будем считать длину построенного пути. Пусть теперь среда изменилась из-за удаления/добавления некоторых препятствий, но расположение начальной и целевой вершин агента осталось неизменным. Задача теперь состоит в поиске пути на модифицированном ГРД.

2 LPLIAN

Наивный подход к поиску пути в условиях динамической среды заключается в повторном вызове алгоритма планирования. В этом случае некоторые выполняемые шаги полностью эквивалентны шагам первого запуска и соответственно некоторые вычисления будут продублированы. Таким образом, целесообразным представляется сохранять промежуточные данные первого запуска и использовать их повторно, с целью экономии вычислительных ресурсов. Lifelong Planning A* (LPA*) – алгоритм, который эффективно использует результаты вычислений, выполненных на первой итерации, для нахождения путей в динамической среде. В этой работе мы применяем подход Lifelong Planning для поиска путей с учетом геометрических ограничений, комбинируя его с алгоритмом LIAN. Результирующий алгоритм назван LPLian. Поскольку LPLian основан на оригинальном алгоритме LIAN и на подходе Lifelong Planning, предоставим информацию об этих двух методах.

2.1 LIAN

LIAN – это эвристический алгоритм поиска, который использует ту же стратегию обхода и обработки состояний, что и классический алгоритм A*. При этом допускаются движения в произвольном направлении (не нарушающие ограничений на угол поворота), также, как и в алгоритме Theta*. Используется идея множественных родителей алгоритма R* [Likhachev and Stentz, 2008].

LIAN, как и любой алгоритм семейства A*, оперирует списками – *OPEN*, в котором располагаются нерассмотренные элементы пространства состояний – кандидаты на дальнейшую обработку; и *CLOSED*, в котором содержатся рассмотренные элементы. В отличие от A*, элемент пространства состояний LIAN идентифицируется не только вершиной графа, то есть ячейкой сетки, но и его родительской вершиной. Это необходимо для учета ограничения на угол поворота. Подобно A*, имеются дополнительные данные, связанные с каждой вершиной (хранятся в памяти во время поиска): g-значение – длина кратчайшего пути к текущей вершине из начальной; *h*-значение – эвристическая оценкой длины кратчайшего пути от текущей вершины к целевой.

На итерации алгоритма происходит извлечение элемента из списка *OPEN*, минимизирующего значение *f*, где $f = g + h$. Этот элемент обрабатывается (раскрывается) следующим образом. Сначала он удаляется из *OPEN* и вставляется в *CLOSED*. Затем создаются элементы-преемники, для этого LIAN использует ячейки, лежащие на заданном входным параметром расстоянии Δ от текущего. Таким образом, LIAN работает только с фрагментами определенной длины, так называемыми Δ-секциями (см. Рис. 1 справа). Вершины-преемники, которые нарушают ограничение достижимости (функция *los*) или ограничение на угол поворота удаляются. Оставшиеся вершины вставляются/обновляются в *OPEN* согласно логике A*. Остановка происходит, если вершина, извлеченная (на очередной итерации) из списка *OPEN*, соответствует целевой клетке графа (в этом случае путь найден) или же если список *OPEN* становится пуст (в этом случае алгоритм возвращает failure, что означает невозможность простроить пути при заданном Δ). В первом случае искомый путь найден и может быть восстановлен с использованием родительских указателей, в противном случае алгоритм сообщает об ошибке, решение не было найдено. На рис.2 слева показан путь, построенный LIAN на пустой сетке со следующими параметрами: $\alpha_{MAX} = 45°, \Delta = 5$.

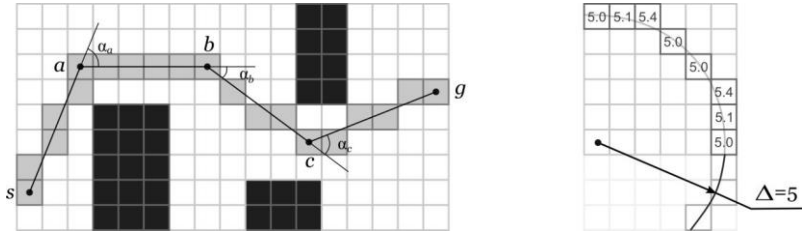


Рис. 1. Граф регулярной декомпозиции и путь на нем (слева), Δ-секция (справа).

2.2 LifeLong Planning

Пусть после того, как путь для данного состояния ГРД был найден, добавляется/удаляется препятствие, при этом начальная и целевая вершины остаются неизменны. Эти изменения влияют на такое свойство вершин как устойчивость (consistency): вершина является устойчивой, если ее g-значение точно равно длине кратчайшего пути в нее из начальной вершины. Если клетка ГРД в процессе появления препятствий становится заблокированной и путь к соответствующей вершине перестает существовать (формально g-значение такой вершины становится $+\infty$), устойчивость пропадает. То же самое верно для клеток, которые стали проходимыми в результате изменений. Более того, клетки, которые находятся рядом с ними могут также утратить свойство устойчивости. В то же время, если удается восстановить устойчивость всех вершин-состояний, второй поиск не потребуется. Это основная идея метода LifeLong Planning [Likhachev et. al., 2001] – восстанавливать устойчивость вершин вместо генерации новых с нуля.

Напрямую применение метода LifeLong Planning применяемого для алгоритма A* к рассматриваемой проблеме планирования траектории с ограничением на угол поворота невозможно, так как пространство поиска в алгоритме LIAN отличается от A*. Описание модификации (метода LifeLong Planning для алгоритма LIAN) представлено далее.

2.3 LPLIAN

Первый запуск LPLian на исходном ГРД ничем не отличается от прохода алгоритма LIAN. Результатом является путь, а также списки *OPEN* и *CLOSED*, заполненные элементами пространства поиска, которые были сгенерированы для построения пути (см. Рис. 2 слева). При обнаружении изменений в графе алгоритм выявляет вершины, потерявшие свою устойчивость, а также генерирует некоторые новые вершины. То есть вместо генерации всех вершин из обоих списков с нуля, обрабатываются только вершины, которые потеряли устойчивость или еще не рассматривались на предыдущем запуске алгоритма. После такой обработки все неустойчивые состояния идентифицированы и помещены в *OPEN* (см. Рис. 2 по центру). Наконец, стандартный цикл алгоритма LIAN возобновляется, то есть вершины извлекаются из списка OPEN и происходит их раскрытие (*expand*) (см. Рис. 2 справа).

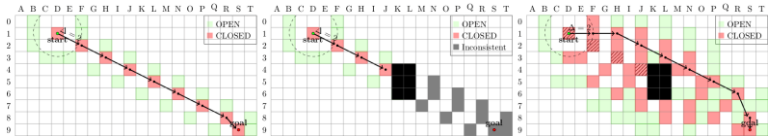


Рис 2. Этапы работы алгоритма LPLIAN (слева-направо): 1) построение пути на первоначальном ГРД, 2) нахождение неустойчивых (отмечены серым) вершин при возникновении препятствия, 3) поиск нового пути с учетом известных списков OPEN и CLOSED.

Опишем этапы работы алгоритма более формально. LPLian использует следующие характеристики вершин: метка вершины g – длина кратчайшего найденного пути в эту вершину, направление движения – определяется вершиной-родителем, метка устойчивости которая задается следующим образом:

$$rhs(a) = \begin{cases} 0, & \text{если} \quad a = start, \\ \min\limits_{a' \in pred(a)} g(a') + cost(a', a), & \text{иначе.} \end{cases}$$

Здесь *cost*(a, b) – вес перехода из a в b, а *pred*(a) – множество вершин, переход из которых в вершину a является допустимым, будем называть их предшественниками вершины a. Вершина a является устойчивой, если выполняется равенство $rhs(a) = g(a)$. Порядок рассмотрения вершин определяется по соответствующему ключу:

$key(a) = [k_1 = \min(g(a), rhs(a) + h(a)), k_2 = \min(g(a), rhs(a))]$.

Вершины добавляются в очередь и извлекаются из нее в порядке минимальности ключа, то есть вершина a должна быть рассмотрена раньше вершины a', если $k_1(a) < k_1(a')$ или $k_1(a) = k_1(a')$ и $k_2(a) < k_2(a')$. Ключ используется такой же, как и в алгоритме LPA*, где его выбор обусловлен следующим фактором: при введении такого ключа алгоритм не требует ручного переключения между состоянием поиска неустойчивых вершин и состоянием стандартной обработки новых вершин. То есть появившаяся неустойчивая вершина будет иметь более высокую приоритетность, чем вершины, для которых она может являться предшественником. Следовательно, если среди ее последователей есть неустойчивые, то они будут найдены на следующем шаге.

Ниже представлен псевдокод алгоритма LPLian, который во многом повторяет этапы алгоритма LPA*. Новизна заключается в добавлении функции *ResetParent*, которая для неустойчивой вершины a находит всех потенциальных предшественников и выбирает из них наилучшего. Особенность заключается в том, что функция ищет вершины, лежащие на расстоянии Δ от текущей вершины и при этом не нарушающие дальнейший угол поворота пути, то есть угол поворота между a и его последователем a'.

Algorithm 1 LPLIAN-Main

```
1: function UpdateVertex(v)
2:     if v ∈ OPEN then
3:         OPEN.remove(v)
4:     end if
5:     if g(v) ≠ rhs(v) then
6:         OPEN.insert(v, key(v))
7:     end if
8: end function
9:
10: function ComputeShortestPath( )
11:     while OPEN.top.key() < key(goal) or rhs(goal) ≠ g(goal) do
12:         current = OPEN.top()
13:         if g(current) > rhs(current) then
14:             g(current) = rhs(current)
15:             expand(current)
16:         else
17:             g(current) = ∞
18:             for v in successors(current) and current do
19:                 ResetParent(v, current)
20:                 UpdateVertex(v)
21:             end for
22:         end if
23:     end while
24:     if g(goal) < ∞ then
25:         return GetPath(goal)
26:     else
27:         return path-not-found
28:     end if
29: end function
30:
31: function FindPath( )
32:     Set all g rhs-values to ∞
33:     rhs(start) = 0
34:     OPEN.insert(start)
35:     while true do
36:         ComputeShortestPath()
37:         wait for the changes
38:         for v: ⟨parent.parent(v), parent(v)⟩ is affected by changes do
39:             ResetParent(v, parent(v))
40:             if parent(v) ≠ NULL then
41:                 UpdateVertex(v)
42:             end if
43:         end for
44:     end while
45: end function
```

Листинг 1. Основной цикл алгоритма LPLIAN.

Более формально функция описывается следующим образом. Изначально, идентифицируется множество потенциальных вершин-кандидатов (множество *U*), которые лежат на Δ-расстоянии от родителя текущей вершины. Затем выбирается узел, удовлетворяющий следующим двум критериям: 1) угол между ячейками ГРД *parent*(*u*), *u*, *parent*, а также угол между *u*, *parent*, *current* удовлетворяют наложенному ограничению; 2) использование сегмента (*u*, *parent*) способствует минимизации длины пути.

Algorithm 2 LPLIAN-ResetParent

```
1: function ResetParent(current, parent)
2:     U - set of nodes at the Δ-distance from parent
3:     for u in U do
4:         node = (parent, u)
5:         if ∠(u, node) < α_max and ∠(node, current) < α_max then
6:             if rhs(parent) > rhs(u) + cost(u, parent) then
7:                 parent(parent) = u
8:                 rhs(parent) = rhs(u) + cost(u, parent)
9:             end if
10:        end if
11:    end for
12:    rhs(current) = rhs(parent) + cost(parent, current)
13: end function
```

Листинг 2. Функция ResetParent

3 Экспериментальные исследования

Для проведения экспериментальных исследований в качестве входных данных использовались фрагменты карт реальной городской местности, полученных из коллекции Openstreetmaps. Всего было использовано 100 фрагментов размером ~ 1,2 на 1,2 км. Указанные фрагменты были преобразованы в ГРД размера 501x501 вершин. Для каждого ГРД было сгенерировано 200 заданий таким образом, что расстояние между начальным и целевым положениями превышало 400 клеток (960 метров).

Одиночный эксперимент состоял в сравнении работы динамической модификации LIAN и нескольких запусков статического LIAN на карте до изменений и после. Тестирование происходило следующим образом, запускалась первая итерация динамического алгоритма, далее карта менялась – на уже построенном пути помещалось препятствие размером 20x10. После изменений, динамический алгоритм LPLian достраивал путь. Для сравнения, на новой карте с препятствием так же запускался статический алгоритм LIAN и подсчитывалось суммарное время работы алгоритмов на обеих картах – 2xLIAN. Так как сложность восстановления пути зависит от того, в каком месте карты произошли изменения, было исследованы различные значения параметра obstacle position, отвечающего за то, на каком расстоянии от клетки старта/финиша было размещено препятствие. Значение параметра obstacle position находится в диапазоне от 0 до 1, и порядковый номер вершины построенного пути, вокруг которой строится препятствие, вычисляется как $i = round(obstacle\_position \times length(path))$. То есть при obstacle position = 0.2 препятствие находится ближе к стартовой вершине, при 0.8 к целевой. Препятствие представляет из себя прямоугольник размера 20x10.

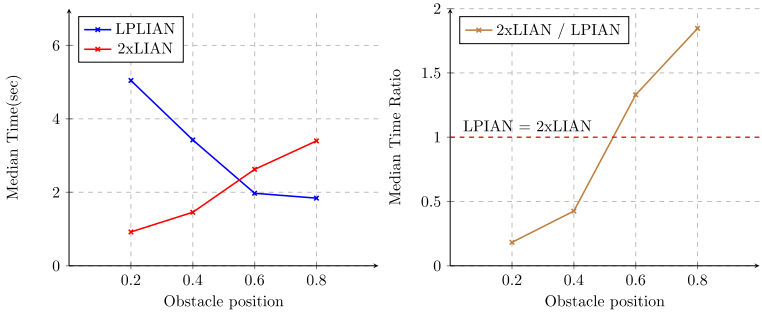


Рис. 5. Графики сравнения работы алгоритмов LPLIAN и двух проходов алгоритма LIAN.

В процессе проведения эксперимента был установлен верхний порог на время выполнения 200 сек, все задания, которые были не решены тем или иным алгоритмом в пределах отведенного времени (470 из 4000) не участвовали в общем усреднении. Сравнивались скорости решения задачи после добавления нового препятствия, то есть время работы стандартного алгоритма LIAN на новой карте с препятствием и время, затраченное на восстановление пути после добавления препятствия для LPLian. Для уменьшения разброса значений, полученных в результате экспериментов, было выбрано медианное значение вместо усредненного. На рисунке 5 (слева) показано взаимное расположение кривых изменения медианы времени работы алгоритмов в зависимости от расположения препятствия и (справа) – изменение отношение времени работы 2xLIAN к LPLIAN. Из графиков видно, что при достаточно близком (0.2. 0.4) расположении препятствия к начальной вершине, алгоритму LPLian требуется больше времени, чтобы перестроить путь, чем построить его просто заново (как это делает 2xLIAN). Это связано с тем, что LPLian совершает два прохода по оставшейся части пути (вершинам, которые потенциально могли потерять устойчивость) – первый проход детектирует неустойчивые вершины, и второй – повторное раскрытие. При перемещении препятствия ближе к целевой вершине время работы LPLian значительно сокращается. А время построения нового пути алгоритмом LIAN становится почти в два раза больше времени, которое тратит LPLian на то, чтобы восстановить путь.

В таблице 1 приведены сравнения медианных значений рассматриваемых выходных параметров обоих алгоритмов. В случае, когда obstacle position = 0.8, получаем выигрыш по времени равный 84.6 %.

Таблица 1. Результаты алгоритмов LPLIAN и двух алгоритмов LIAN.

obstacle position	0.2	0.4	0.6	0.8
LPLIAN	5.045	3.425	1.974	1.842
2xLIAN	0.921 (- 81.7 %)	1.457 (- 57.5 %)	2.626 (+ 33 %)	3.4 (+ 84.6 %)

3 Заключение

Использование предложенного алгоритма оправдано при навигации в динамической среде, если изменения происходят ближе к целевой вершине. Это особенно важно, т.к. при планировании траектории в ограниченно-наблюдаемой среде обычно старт и цель меняют местами исходя из алгоритмических соображений и почти все изменения происходят рядом с агентов (т.е. у цели).

Список литературы

[Андрейчук и др., 2017] Андрейчук А.А, Яковлев К.С. Методы планирования траектории на плоскости с учетом геометрических ограничений // Известия РАН. Теория и системы управления, 2017. № 6, с. 125–156.

[Яковлев, 2009] Яковлев К.С. Графы специальной структуры в задачах планирования траектории. Труды III международной конференции «Системный анализ и информационные технологии САИТ-2009». М: ИСА РАН, 2009.

[Яковлев и др., 2013] Яковлев К.С., Баскин Е.С. Графовые модели в задаче планирования траектории на плоскости // Искусственный интеллект и принятие решений. 2013. №1. С. 5-12.

[Daniel et al., 2010] Daniel K., Nash A., Koenig S., Felner A. Theta*: Any-angle Path Planning on Grids // J. Artificial Intelligence Research. 2010. V. 39. P. 533–579.

[Harabor and Grastien, 2011] Harabor, D.D., Grastien, A.: Online graph pruning for pathfinding on grid maps. In: Proceedings of the 25th AAAI Conference on Artificial Intelligence (AAAI-2011). pp. 1114–1119 (2011)

[Hart et. al., 1968] Hart, P.E., Nilsson, N.J., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. IEEE transactions on Systems Science and Cybernetics, 4(2), 100–107 (1968)

[Likhachev et. al., 2001] Likhachev, M., Koenig, S.: Lifelong planning A* and dynamic A*lite: The proofs. In: Technical report, College of Computing, Georgia Institute of Technology, Atlanta (Georgia) (2001)

[Likhachev and Stentz, 2008] Likhachev, M., Stentz, A.: R* search. In: Proceedings of the 23rd AAAI Conference on Artificial Intelligence, AAAI-2008. pp. 344–350 (2008)

[Yap, 2002] Yap, P.: Grid-based path-finding. In: Proceedings of 15th Conference of the Canadian Society for Computational Studies of Intelligence. pp. 44–55. Springer (2002)

УДК 004.896

ОБ ОДНОМ ВОПРОСЕ РЕАЛИЗАЦИИ АЛГОРИТМА ПЛАНИРОВАНИЯ ТРАЕКТОРИИ A*

С.А. Дергачев (sadergachev@edu.hse.ru)
МИЭМ им. Тихонова НИУ ВШЭ, Москва

К.С. Яковлев (kyakovlev@hse.ru)
ФИЦ ИУ РАН, Москва
НИУ ВШЭ, Москва
МФТИ, Долгопрудный

Аннотация. В работе рассматривается задача планирования траектории агента (мобильного робота, беспилотного транспортного средства и др.) в статической среде. В мобильной робототехнике требуется получать решение этой задачи за ограниченное время, что может быть затруднительным в условиях использования малопроизводительного аппаратного обеспечения. В работе исследуется вопрос организации хранения результатов промежуточных вычислений при планировании траектории с помощью алгоритма A* для повышения его быстродействия. Предлагаются различные варианты реализации алгоритма, приводятся результаты экспериментального исследования.[1]

Ключевые слова: планирование траектории, A*, эвристический поиск.

Введение

Для повышения автономности беспилотных транспортных средств необходима разработка интеллектуальных систем управления, важным компонентом которых является модуль автоматического планирования траектории [Макаров и др., 2015]. Зачастую планирование траектории сводится к поиску пути на графе, вершины которого соответствуют некоторым положениям объекта в пространстве, а ребра – простейшим траекториям между ними. Для решения задачи в такой постановке зачастую используются эвристические алгоритмы семейства A* [Hart et al., 1968] такие как Theta* [Daniel et al., 2010], LIAN [Yakovlev, 2015] и др. Многие алгоритмы этого семейства обладают важными теоретическими

[1] Работа выполнена при частичной поддержке РФФИ (проекты №№ 17-07-00281, 18-37-20032)

свойствами (полнота и оптимальность), но на практике важна также скорость работы, особенно – в контексте применения на маломощных вычислителях, широко используемых в мобильной робототехнике, таких как, например, Raspbery Pi [Андрейчук и др., 2016]. Таким образом, актуальной является задача повышения быстродействия алгоритмов планирования за счет оптимизаций при их реализации. Именно этой проблеме и посвящена настоящая работа.

Для обеспечения корректного функционирования алгоритмам эвристического поиска семейства A* требуется хранить большой объем результатов промежуточных вычислений и обеспечивать быстрый поиск элементов, при этом индексация по фиксированному ключу невозможна, т.к. в различных ситуациях требуется поиск по различным характеристикам. Таким образом, для определения наиболее эффективного (с точки зрения быстродействия) способа организации хранения данных необходимо: а) программно реализовать несколько вариантов метода, совпадающих алгоритмически, но отличающихся используемыми структурами хранения данных (контейнерами); б) провести экспериментальное исследование полученных реализаций. Именно эти задачи и были решены в ходе данной работы.

1 Постановка задачи планирования

Рассмотрим задачу планирования траектории мобильного робота на плоскости как задачу поиска пути на графе особой структуры – графе регулярной декомпозиции, метрическом топологическом графе (ГРД/МТ-граф) [Яковлев и др., 2013]. Этот граф может быть представлен в виде матрицы меток «0» и «1» («свободно», «заблокировано») или таблицы, состоящей из заблокированных (непроходимых) и свободных (проходимых) ячеек квадратной формы. Упомянутая таблица получается наложением регулярной сетки на рабочую область робота и закрашиванием тех ячеек, в которых содержится хотя бы одна непроходимая для робота точка. Центры ячеек – вершины графа. Ребра соответствуют парам смежных проходимых вершин. Вес ребра – эвклидово расстояние между вершинами (центрами клеток). Задача состоит в поиске кратчайшего пути на таком графе.

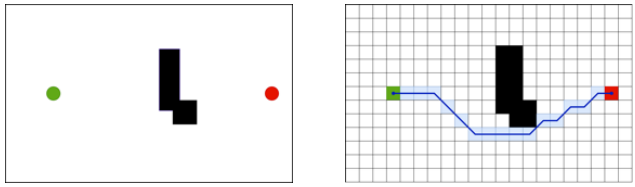


Рис. 1. Пример представления пространства в качестве графа особой структуры

2 Алгоритм A* и вопросы его реализации

Алгоритм A* является эвристической модификацией алгоритма Дейкстры [Dijkstra, 1959]. Для поиска пути осуществляется итеративный обход графа, при этом используется эвристика, влияющая на то, какую вершину обрабатывать на очередном шаге. Алгоритм выбирает для обработки вершину минимизирующую функцию $f(v) = g(v) + h(v)$, где $g(v)$ – это вес пути из начальной вершины в вершину v, известный к текущей итерации, а $h(v)$ – это оценка веса пути из v в целевую вершину. С алгоритмической точки зрения можно считать, что h-значения всех вершин известны (или вычисляются за константное время, которым можно пренебречь). g-значения, наоборот, изначально неизвестны для всех вершин, кроме начальной, а на уровне псевдокода полагают эти значения равными бесконечности.

После того как на текущей итерации выбрана вершина минизирующая $f(v)$, например v', происходит её раскрытие (expansion), в ходе которого производится перебор всех смежных с v' вершин – v'', и сравнение $g(v'')$ и $g(v') + w(v', v'')$, где w – вес соответствующего ребра. Если $g(v') + w(v', v'')$ меньше, чем $g(v'')$, то есть найден более короткий путь к v'', то, во-первых этот факт фиксируется, а именно $g(v'')$ полагается равным $g(v') + w(v', v'')$, во-вторых для вершины v'' запоминается ссылка на v', то есть теперь вершина v'' является родительской для вершины v''. С использованием этих ссылок происходит восстановление пути после окончания расчёта g-значений. Это окончание происходит, когда к раскрытию выбирается целевая вершина.

Псевдокод алгоритма A* представлен на Рис.2. Предполагается, что для организации обхода графа используются два контейнера – OPEN и CLOSE. OPEN – контейнер, содержащий вершины-кандидаты, из которых выбирается вершина для раскрытия на текущей итерации алгоритма. Изначально это список содержит лишь начальную вершину и пополняется за счет добавления в него вершин, смежных с раскрываемой. CLOSE – контейнер для раскрытых вершин. Заметим, что при использовании эвристики, удовлетворяющей условиям монотонности, не требуется перераскрытие вершин для нахождения кратчайшего пути [Pearl, 1984], таким образом CLOSE может быть использован так, как указано в строке 17 псевдокода: если вершина смежная с раскрываемой содержится в CLOSE, то перерасчёт её g-значения не требуется (строки 18-20 пропускаются).

Алгоритм 1 Алгоритм A^*

```
1: Open := {}
2: Close := {}
3: g(v_start) := 0
4: for v ∈ V do
5:     g(v) := inf
6:     parent(v) := NIL
7: end for
8: Open := Open ∪ {v_start}
9: while Open ≠ ∅ do
10:    v' = argmin_{v ∈ Open}(F(v))
11:    Open := Open\{v'}
12:    Close := Close ∪ {v'}
13:    if v' = v_goal then
14:        break
15:    end if
16:    for v'' ∈ successors(v') do
17:        if v'' ∉ Close and g(v'') > g(v') + w(v', v'') then
18:            g(v'') := g(v') + w(v', v'')
19:            parent(v'') := v'
20:            Open := Open ∪ {v''}
21:        end if
22:    end for
23: end while
24: if v_goal ∉ Close then
25:    print("Path not found")
26: else
27:    GetPathFromParents()
28: end if
```

Рис. 2. Псевдокод алгоритма A*

Указанный на Рис. 2 псевдокод, являясь алгоритмически корректным, опирается на ряд допущений, реализация которых на практике сопряжена с определенными вопросами. Во-первых, строки 4-7 неявно предполагают генерацию структур данных, необходимых для поиска и соответствующих *каждой* вершине графа. Назовём такую структуру Node. Очевидно, что хранение в памяти данных обо всех вершинах графа для конкретной задачи избыточно (часто в процессе поиска рассматривается лишь малый процент от всех вершин). Во-вторых, строка 10 подразумевает поиск Node с минимальным *f*-значением среди всех Node, принадлежащих OPEN. На практике это означает, что для быстрого выполнения этого шага необходима индексация OPEN по ключу *f*. С другой стороны, в строке 17 неявно подразумевается поиск в OPEN вершины, смежной с текущей раскрываемой. Для быстрого выполнения этого шага необходима индексация по идентификатору вершины. В рассматриваемом случае, когда речь идёт о поиске пути на MT-графе, этот идентификатор есть пара индексов вершины *i* и *j* (или их свертка до одного целого числа). Таким образом, на практике к контейнеру OPEN выдвигаются противоречивые требования – с одной стороны быстрый поиск по идентификатору, с другой – быстрый поиск минимума по *f*-значению. Псевдокод более приближенный к практической реализации, акцентирующий внимание на указанных вопросах представлен на Рис. 3.

Теперь в процессе выполнения граф поиска (множество структур Node, соответствующих всем вершинам исходного графа) не хранится целиком в оперативной памяти, и новые элементы пространства поиска, Node, генерируются по мере надобности. Каждый элемент инкапсулирует идентификатор вершины (координаты клетки MT-графа), её *h*-значение, *g*-значение, индексы вершины *i* и *j*, а также ссылку на родительскую вершину (parent). Операции поиска/добавления в OPEN и CLOSE теперь прописаны явно.

Алгоритм 3 Алгоритм *A** (модификация)

```
1: Open := new OpenStructure()
2: Close := new CloseStructure()
3: N_start := new Node(v_start)
4: N_goal := new Node(v_goal)
5: N_start.g := 0
6: Open.AddOrUpdate(N_start)
7: while Open.Size ≠ 0 do
8:     N' = Open.GetOptimal()
9:     Close.Add(N')
10:    if N' = N_goal then
11:        return GetPathFromParents()
12:    end if
13:    for N" in N'.GetSuccessors() do
14:        if not Close.Contain(N") then
15:            N".g := N'.g + w(N', N")
16:            N".parent() := N'
17:            Open.AddOrUpdate(N")
18:        end if
19:    end for
20: end while
21: return PathNotFound()
```

Рис. 3. Псевдокод, описывающий программную реализацию алгоритма A*

Таким образом, при практической реализации алгоритма A* (или *любого* другого алгоритма этого семейства) возникает важный вопрос – каким образом организовать хранение и упорядочивание элементов поиска (Node) в OPEN и CLOSE? От CLOSE требуется лишь быстрый поиск по идентификатору, то есть для программной реализации можно однозначно рекомендовать использование хэш-таблицы, вставка и поиск по которой осуществляются за $O(1)$. Требования же к контейнеру OPEN, как было сказано, противоречивы. С одной стороны необходим быстрый поиск минимума по f (строка 8), с другой – быстрый поиск по идентификатору для добавления/обновления (строка 17). Удовлетворить обеим требованиям одновременно невозможно, поэтому предлагается реализовать OPEN с использованием различных контейнеров данных и провести экспериментальное исследование.

Рассмотрим подробнее контейнеры, которые были выбраны для сравнения.

Линейный список (list) – структура, хранящая упорядоченный набор данных с последовательным доступом к элементам. Хранение вершин в списке было реализовано двумя способами: с сохранением дубликатов (т.е. вершин с совпадающими индексами *i* и *j*) и без сохранения (w/ duplicates и w/o duplicates соответственно). Хранение дубликатов не влияет на теоретические свойства алгоритма, но, очевидно, ускоряет добавление элемента (поиск по идентификатору вершины просто не производится). Для быстрого нахождения вершины с минимальным *f*-значением, элементы списка упорядочены по нему. При совпадении *f*-значений вершин дальнейшее упорядочивание производилось по g-значению, индексу *i*, индексу *j*. Таким образом, можно говорить о составном ключе (*f, g, i, j*), который однозначно задает линейный порядок на множестве элементов поиска. Аналогичная система упорядочивания вершин использовалась и для программных реализаций с другими контейнерами.

Упорядоченное множество (set) – структура, хранящая упорядоченный набор данных, которая исключает хранение эквивалентных элементов. Аналогично списку используется в вариантах w/ duplicates и w/o duplicates (за исключением элементов с полностью ссыпающим набором свойств (f, g, i, j), т.к. они будут эквивалентны).

Очередь с приоритетами (priority queue) – структура данных, поддерживающая операции добавления нового элемента и нахождения элемента с максимальным приоритетом. Внутреннее устройство этого контейнера не предусматривает доступ к элементам, за исключением элемента с наибольшим приоритетом, а значит невозможно произвести операцию поиска и удаления дубликатов вершин, таким образом очередь с приоритетами реализована только в варианте w/duplicates.

Динамический массив (vector), элементами которого являются экземпляры вышеописанных контейнеров (например: vector of lists w/ duplicates). Индексация массива осуществлялась по *i* индексам вершин. То есть каждый элемент массива хранит экземпляры определенного контейнера, в котором содержаться лишь вершины с одинаковым *i*. Такая индексация удобна при использовании MT-графов, т.к. значение *i* меняется от 0 до заранее известного предела (высоты карты в клетках). Благодаря такой организации, скорость доступа к вершинам по идентификатору существенно повышалась, но несколько замедлялся поиск элемента с наименьшим *f*.

Таким образом, для дальнейших исследований была подготовлена реализация алгоритма A*, с возможностью выбора одного из 10 вышеописанных контейнеров.

3 Экспериментальное исследование

Для проведения экспериментальных исследований алгоритм A* был реализован на языке C++ в соответствие с указанным псевдокодом (см. рис. 2). В качестве контейнеров OPEN были использованы реализации выбранных структур из библиотеки стандартных шаблонов (STL) C++. Эксперименты проводились на двух типах карт – синтетических (пустых, искусственно сгенерированных, с изменяющимся размером) и реальных (взятых из открытых коллекций данных или построенных по реальным картам городской местности). Исследование с использованием синтетических карт производилось на компьютере на базе Intel Core i7-3615QM @2.3 ГГц, 8 Гб ОЗУ под управлением Windows 10 (64-bit). Тестирование на реальных картах производилось на компьютере с Windows 7 (32-bit) на базе Intel Core 2 Duo Q8300 @2.5Ghz, 2 Гб ОЗУ. Исходный код реализации алгоритма доступен на https://github.com/PathPlanning/AStar-DCO.

3.1. Тестирование на синтетических картах

Для проведения экспериментальных исследований генерировались пустые карты разного размера с стартом и финишем расположенными в диагонально противоположенных углах карты. Размер карт менялся от 5000 x 5000[2] до 20 000 x 20 000 с шагом в 100 единиц. При подобном подходе количество итераций алгоритма, требуемых для выполнения задания, растет линейно. На каждой итерации при этом проверяется одно и тоже число смежных вершин, т.к. отсутствуют препятствия. Таким образом, такой подход позволяет линейно увеличивать сложность заданий, т.к. общее время выполнения зависит в основном от времени, которое требуется на работу с контейнером OPEN (добавление, поиск по идентификатору, поиски минимума по *f*-значению).

Одиночный эксперимент состоял в запуске алгоритма A* с различными реализациями контейнера OPEN для отдельного задания. Поскольку на время выполнения оказывают влияние процессы, запущенные на ЭВМ параллельно этой задаче, одно и тоже задание для каждого контейнера запускалось 5 раз, а потом бралось среднее арифметическое времени выполнения нескольких запусков. График, демонстрирующий зависимость этих времен от типа используемого контейнера OPEN приведен ниже (Рис. 4).

[2] Для карт с размером менее 5 000 x 5 000 время работы алгоритма было меньше 10^{-3} и не поддавалось точному измерению, поэтому такие задания не рассматривались.

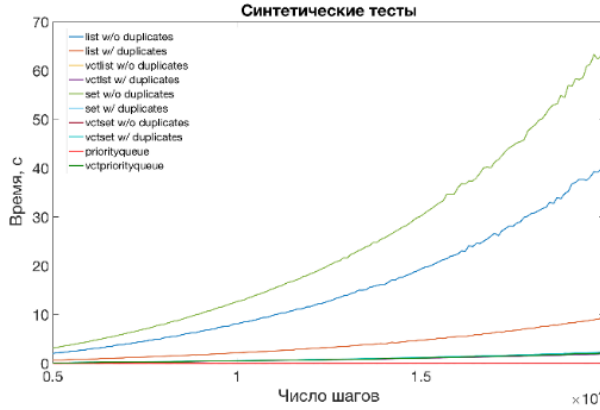


Рис. 4. График зависимости времени выполнения теста от количества шагов для всех контейнеров

Можно видеть, что наименьшее быстродействие показывают контейнеры: set без дубликатов, list без дубликатов и list с дубликатами. Для контейнеров с большим быстродействием был построен следующий график (Рис. 5). На нем видно, что priority queue и set с дубликатами лидируют по быстродействию Следующими по скорости являются векторы list с дубликатами и без дубликатов. Их результаты почти полностью совпадают, но заметно отстают от set с дубликатами и priority queue.

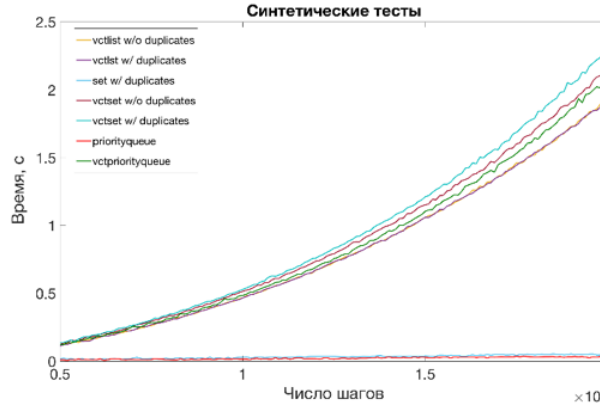


Рис. 5. График зависимости времени выполнения теста от количества шагов для 7 первых контейнеров

3.2 Тестирование на реальных картах

Во второй части эксперимента были использованы три набора из 36, 75 и 100 карт соответственно. Первые два набора – Warcraft и Baldur's Gate – состоят из карт размером 512 на 512 ячеек, использующихся в игровой индустрии [Sturtevant, 2012]. Для них существуют банк заданий, для каждого из которых известна длина кратчайшего пути. Задания для каждой карты были отсортированы по этой длине и было выбрано по 100 заданий с самыми длинными путями. Третий набор состоял из фрагментов карты Москвы, полученных из открытой гео-информационной базы данных OpenStreetMaps (www.openstreetmap.org) размером 501 на 501 ячейку. Меняя расположение ячеек начала и конца пути для каждого фрагмента было составлено по 100 различных заданий. Ячейки начала и конца выбирались таким образом, чтобы они располагались на противоположенных сторонах карты с отступом 10 ячеек от края. Стороны карты и неопределенные координаты выбирались случайным образом. Для каждого задания было проведено по 6 запусков реализации алгоритма и было взято среднее значение времени выполнения.

При анализе результатов было обнаружено, что использование различных контейнеров по-разному влияет на быстродействие алгоритма в зависимости от числа итераций основного цикла. Так, усредняя по всем заданиям получается график, представленный на рис. 7, а при усреднении по заданиям, для решения которых требуется 4 000 шагов и менее, получается результаты представлены на рис. 6. Основное отличие в том, что для таких заданий варьируется время работы для «средних» контейнеров (т.е. не самых быстрых и не самых медленных) в зависимости от набора карт. При усреднении по всем заданиям картина более однородная.

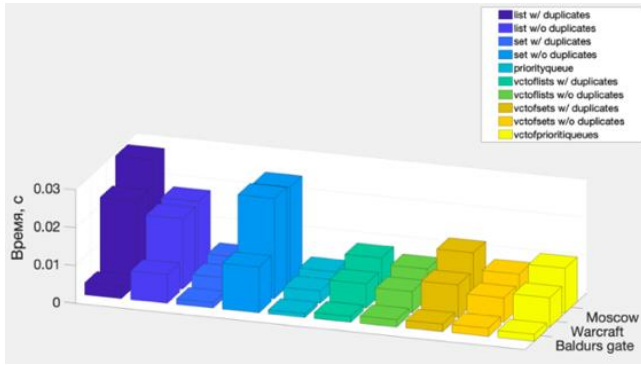


Рис. 6. Среднее время работа на заданиях с малым (<4 000) числом шагов.

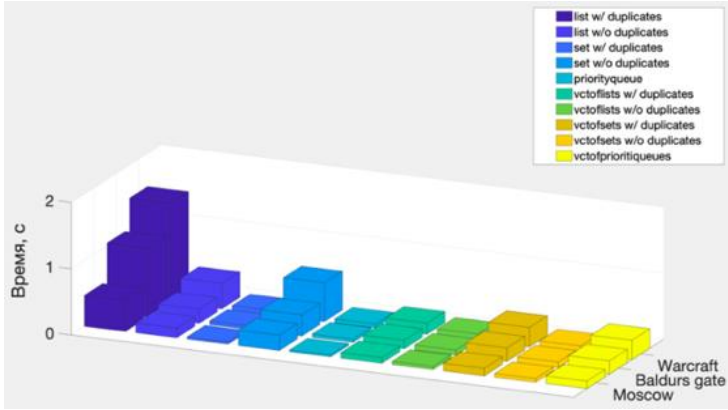


Рис. 7. Время работы, усредненное по всем заданиям.

3.3 Выводы

По результатам экспериментального исследования можно сделать следующие выводы. Во-первых, использование различных контейнеров существенно влияет на производительность (скорость работы) алгоритма. Таким образом, очень важно с практической точки зрения использовать подходящие контейнеры данных при реализации алгоритма планирования. Во-вторых, контейнеры с дубликатами всегда быстрее, чем контейнеры без них. Однако, очевидно, что их использование ведёт к дополнительным затратам памяти. Поэтому в приложениях, где критична как скорость работы, так и затраты памяти, рекомендуется использовать вектор списков или вектор множеств.

Заключение

В работе был рассмотрен ряд вопросов, касающихся практической реализации алгоритмов эвристического поиска семейства A*, обычно использующихся в мобильной робототехнике для планирования траектории. Основное внимание было уделено вопросу выбора программного контейнера для хранения результатов промежуточных вычислений. Была продемонстрирована критическая зависимость быстродействия алгоритма от выбранного способа хранения (контейнера). В результате проведения масштабных экспериментальных исследований различных контейнеров даны рекомендации по их использованию для создания высокоэффективных (в смысле скорости работы) реализация алгоритмов планирования.

Список литературы

[Андрейчук и др., 2016] Андрейчук А. А. Боковой А. В. Яковлев К. С. Оценка быстродействия некоторых алгоритмов планирования траектории на широко используемой в робототехнике платформе Raspberry PI // Экстремальная робототехника – 2016 – №1 – С. 184-189.

[Макаров и др., 2015] Макаров Д.А., Панов А.И., Яковлев К.С. Архитектура многоуровневой интеллектуальной системы управления беспилотными летательными аппаратами // Искусственный интеллект и принятие решений, 3, 2015. С.18-32.

[Яковлев и др., 2013] Яковлев К.С., Баскин Е.С. Графовые модели в задаче планирования траектории на плоскости // искусственный интеллект и принятие решений, 1, 2013. С.5-12.

[Daniel et al., 2010] Daniel, K., Nash, A., Koenig, S., and Felner, A. Theta*: Any-angle path planning on grids. Journal of Artificial Intelligence Research. 2010. No 39, P.533-579.

[Dijkstra, 1959] Dijkstra, E.W. A note on two problems in connexion with graphs. Numerische Mathematik 1(1), 1959. Pp 269–271.

[Hart et al., 1968] Hart, P. E., Nilsson, N. J., and Raphael, B. A formal basis for the heuristic determination of minimum cost paths. IEEE transactions on Systems Science and Cybernetics. 1968. No 4(2), P.100-107.

[Pearl, 1984] Pearl J. Heuristics: Intelligent Search Strategies for Computer Problem Solving. Addison-Wesley, 1984

[Sturtevant, 2012] Sturtevant N. R. Benchmarks for grid-based pathfinding //IEEE Transactions on Computational Intelligence and AI in Games. – 2012. – V. 4. – №. 2. – pp. 144-148.

[Yakovlev, 2015] Konstantin Yakovlev, Egor Baskin, and Ivan Hramoin. Grid-based angle-constrained path planning. In Proceedings of The 38th Annual German Conference on Artificial Intelligence (KI'2015), pages 208-221, 2015. Springer International Publishing.

УДК 62-52, 001.57, 004.358, 004.942, 519.876.5

РАЗРАБОТКА НЕЙРОЭВОЛЮЦИОННОГО КОНТРОЛЯ МАШИНЫ С РУЛЕВЫМ УПРАВЛЕНИЕМ АККЕРМАНА В СИМУЛЯТОРЕ V-REP

Д.Ю. Ларионова (*d.larionova@innopolis.ru*)
М.А. Иванов (*m.ivanov@innopolis.ru*)
И.М. Афанасьев (*i.afanasyev@innopolis.ru*)

Университет Иннополис, Иннополис

Аннотация. Разработка интеллектуального контроля для транспортных систем является перспективной задачей современной мобильной робототехники. В этой статье разработан нейроэволюционный контроль машины с рулевым управлением по схеме Аккермана, работа которого продемонстрирована в симуляторе V-REP. Для моделирования эксперимента были разработаны 3D сцена, сценарий, модель машины, учитывающая кинематические особенности движения аккермановской машины. Для автоматического тестирования были созданы модули управления машины в среде Python Remote-API и контроля схода машины с трассы. Сперва в симуляции был исследован ПИД-контроль движения машины вдоль линии на основе показаний видеосенсоров, затем для контроля и стабилизации машины на трассе был разработан нейроэволюционный алгоритм NEAT с входными данными, как от видеосенсоров, так и от сенсоров расстояния. Работа алгоритма была проверена на разных трассах, показав, что нейроэволюционный контроль успешно справляется с управлением аккермановской машины, формируя оптимальные параметры контроля (углы рулевого управления и скорости) и оптимальную структуру нейросети для управления машиной.

Ключевые слова: нейроэволюционный контроль, рулевое управление машины, схема Аккермана, NEAT алгоритм, ПИД-контроллер, симулятор V-REP, моделирование движения

Введение

Разработка интеллектуального контроля для транспортных систем является перспективной задачей современной мобильной робототехники, требующей междисциплинарного подхода. Анализ литературы показывает, что движение мобильных объектов часто изучается с использованием

моделей роботов в 3D симуляторах, таких как Gazebo [Афанасьев и др., 2015; Sokolov et al., 2016, Shimchik et al., 2016; Sokolov et al., 2017a], MATLAB [Лавренов и др., 2016; Magid et al., 2017], MATLAB/Simulink [Khusainov et al., 2016], Scilab [Dobriborsci et al., 2016], V-REP [Rohmer et al., 2013], применение среды ROS/RViz [Зенкевич и др., 2017; Ibragimov et al., 2017; Bokovoy et al., 2018; Filipenko et al., 2018; Лавренов и др., 2019], гоночных симуляторов, таких как Speed Dreams [Zubov et al, 2018], а также прикладного программного обеспечения с генерацией антропоморфных моделей [Gabbasov et al., 2015; Danilov et al., 2016]. Кроме того, подобные симуляторы используются для генерации карт и разработки специальных сред [Afanasyev et al., 2015; Lavrenov et al., 2017]. Движение машин с аккермановскими схемами рулевого управления исследовалось в работах [Shimchik et al., 2016; Sheikh et al., 2018; Filipenko et al., 2018; Zubov et al, 2018]. Использование нейроэволюционных алгоритмов для управления роботами представлены в исследованиях [Ahmadzadeh et al., 2015; Sokolov et al., 2017b]. В данной работе предложен и реализован в симуляторе V-REP нейроэволюционный метод управления аккермановской машины, формирующий команды контроля углов рулевого управления и скорости.

1 Процесс моделирования эксперимента в V-REP

1.1 Моделирование эксперимента с машиной в среде V-REP

В качестве среды моделирования эксперимента с аккермановской машиной использовался симулятор V-REP компании Coppelia Robotics. Из библиотеки V-REP была применена модель аккермановской машины (Рис. 1), с установленными на ней тремя видеосенсорами, направленными в пол для контроля интенсивности света и тремя датчиками близости (proximity sensors). В среде V-REP специально для эксперимента была создана гоночная трасса полигона (Рис. 1), состоящая из стен ограждения и дороги, с линией замкнутой формы посередине, для движения машины по замкнутой траектории. Треки дороги имеют разную форму, но общий стиль. Черная линия, вдоль которой движется автомобиль, окружена стенами на равном удалении, которые также используются для контроля положения машины внутри полигона (по равноудаленности от стен по датчикам приближения). Красные линии нужны для автоматического определения схода машины с дистанции (по видеосенсорам). Таким образом, был создан сценарий в среде V-REP в формате *.ttt, реализующий 3D сцену и модель аккермановской машины (с шасси, приводами, рулевым управлением и сенсорами), полностью отражающий кинематические особенности движения машины.

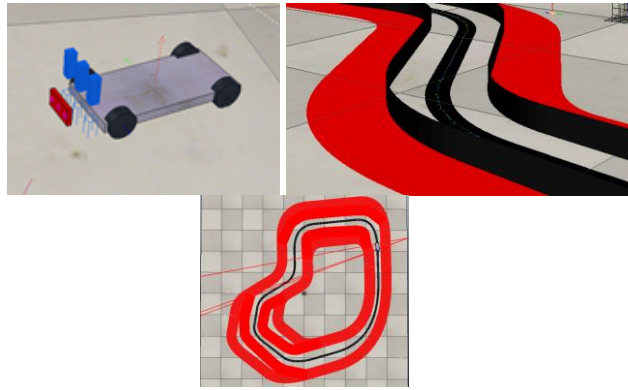


Рис 1. 3D модель машины с рулевым управлением по схеме Аккермана (слева). Смоделированная гоночная трасса полигона из стен ограждения и дороги замкнутой формы для эксперимента в симуляторе V-REP (в центре и справа)

1.2. Разработка алгоритма стабилизации машины с помощью ПИД-контроллера в среде V-REP

Задача заключалась в симуляции движения аккермановской машины вдоль линии (line following) со стабилизацией движения при помощи ПИД-контроллера. При симуляции машина движется по трассе полигона с постоянной скоростью (Рис. 2). Для определения положении автомобиля относительно линии движения считываются значения интенсивности света с левого и правого видеосенсоров и рассчитывается разность интенсивности света между ними. В условиях симуляции показания видеосенсора варьируются в диапазоне от 0 (черный цвет, минимальная освещенность) до 1 (белый цвет, максимальная освещенность). Если линия проходит строго между двумя сенсорами, в идеальных условиях их показания будут равны (разность интенсивности света будет равна 0). На основе разности показаний датчиков рассчитывается ПИД-коэффициенты. устанавливающие угол рулевого управления (отрицательные и положительные значения разности показаний), что в конечном счете определяет направление поворота автомобиля. Псевдокод и блок-схема алгоритма контроля движения машины показаны на Рис. 3.

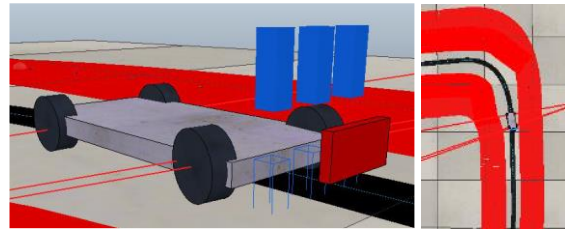


Рис 2. Симуляция эксперимента с движением машины по трассе полигона и трекингом линии сенсорами в симуляторе V-REP

```
var prev_error = 0, error, integral = 0, derivative = 0,
left_intensity, right_intensity, computed_angle
        while true
        do
                left_intensity = read_left_sensor_data ()
                right_intensity = read_right_sensor_data ()
                error = right_intensity - left_intensity
                integral = integral + error
                derivative = error - prev_error
                computed_angle =
                deg_to_radians(kp*error + ki*integral +
                kd*derivative)
                set_steering_angle(computed_angle)
        end
```

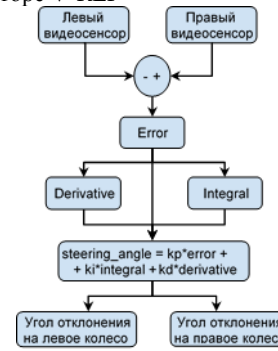


Рис 3. Псевдокод и блок-схема алгоритма контроля движения машины вдоль линии на основе показаний видеосенсоров и ПИД-контроллера

2 Разработка алгоритма нейроэволюционного контроля машины с аккермановским управлением в среде V-REP

2.1. Разработка алгоритма управления машины в V-REP

На данном этапе был разработан алгоритм, написанный в Lua-скрипте V-REP сцены. Он реализовал контроль движения машины вдоль линии на основе показаний видеосенсоров и ПИД-контроллера (см. раздел 1.2). Проект содержал всего один файл сцены V-REP для 3D модели машины с аккермановским рулевым управлением в формате ttt-файла (*.ttt). С одной стороны, этот подход удобен, поскольку запуск требует только установку V-REP и загрузку сценария (ttt-файла) с полигоном и смоделированной машиной. Так что для старта проекта не требуются дополнительные компиляторы или интерпретаторы, достаточно запустить стартовую команду в симуляторе V-REP. С другой стороны, такой подход не практичен, так как добавление новых особенностей (features) требует временных затрат для оптимизации. Поэтому потребовалась новая реализация управления аккермановской машины с последующей автоматизацией алгоритма подбора параметров. При анализе интеграции

симулятора V-REP с другими языками программирования выбор пал на Python Remote-API. В результате этого этапа, к ttt-файлу в проект V-REP добавилось еще несколько файлов (помимо API): main.py, connections.py, и AckermannCar.py. Описание назначения файлов приведено в Таблице 1. У данного программного решения оказался только один недостаток, на тот момент никак себя не проявивший: класс AckermannCar был выполнен для алгоритма следования линии, что исключало возможность для управления машиной во время симуляции другими способами.

Таблица 1. Структура проекта с контролем машины по схеме Аккермана в V-REP с управлением на основе Python Remote-API

Файлы проекта	**Назначение**
main.py	Стартовый скрипт с параметрами запуска проекта
connections.py	Скрипт с методами установления и закрытия соединения с V-REP по API, загрузкой сцены и запуска симуляции
AckermannCar.py	Скрипт с описанием одноименного класса, конструктор которого принимает 4 параметра (помимо ID соединения для вызова функций V-REP): скорость автомобиля и 3 коэффициента ПИД-контроллера. Метод run_car() содержит алгоритм, описанный в псевдокоде на Рис. 3.

Ручной процесс подбора параметров неэффективен и необъективен, поскольку может упустить из виду лучшие варианты. Поэтому появилась необходимость автоматизации тестирования с различными стартовыми параметрами. На этом этапе в проект был добавлен файл TestDataGenerator.py, который инициализировался массивом с возможными диапазонами и шагом изменения генерируемых параметров. Так, при каждом вызове get_params() генерировалось уникальное сочетание параметров, которые записывались в файл *.csv с соответствующим временем симуляции. Однако, как определить, что машина все еще следует линии? Для этого было принято решение обвести существующую трассу с обеих сторон на некотором расстоянии красными линиями, пересечение с которыми означало бы сход автомобиля с дистанции (Рис. 1). Для регистрации факта схода с трассы использовался средний видеосенсор на машине для распознавания изменения цветов (Рис. 2). Это решение оказалось удачным и использовалось для экспериментов и набора статистики прохождения трассы машиной.

2.2. Разработка нейроэволюционного алгоритма для управления машиной в симуляторе V-REP

Хотя ПИД-контроллер надежно контролирует движение машины, целью исследования является разработка нейроэволюционного алгоритма стабилизации машины на трассе. Чтобы использовать для управления автомобилем нейросеть требуется её обучить (т.е. подобрать коэффициенты), что не представляет проблему, если известна топология сети. Однако, составление архитектуры сети является сложной задачей. Поэтому была применена библиотека эволюционных алгоритмов NEAT (NeuroEvolution of Augmenting Topologies), помогающая подобрать топологию нейросети [Stanley et al, 2002]. Библиотека NEAT совместима с Python Remote-API, что важно для нашей симуляции в V-REP.

Принцип работы алгоритма следующий (Рис. 4). На вход подаются данные с трех сенсоров одного вида (видео или расстояния), на выходе получаем угол рулевого управления и скорость машины. Чтобы заменить в симуляции ПИД-контроллер на нейроэволюционное управление понадобился рефакторинг проекта. Теперь в классе AckermannCar остались только методы чтения данных сенсоров и изменения скорости и угла рулевого управления. Алгоритм следования линии был вынесен в отдельный класс-контроллер, что позволило быстро менять алгоритм управления машины (ПИД-контроллер или нейросеть) и модуль исследуемой задачи (следование линии, автономная навигация или др.).

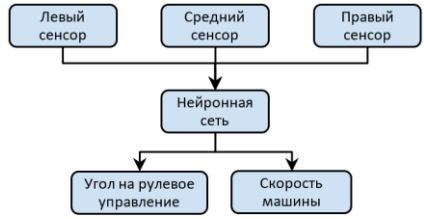


Рис 4. Симуляция эксперимента с движением машины по трассе полигона и трекингом линии сенсорами в симуляторе V-REP

3 Результаты работы алгоритма нейроэволюционного контроля машины в симуляторе V-REP

Сперва алгоритм тестировался на сцене ackerman_car.ttt. При одних и тех же начальных настройках геномы-долгожители (достигшие лимита по времени симуляции) появились уже на ранних поколениях при использовании видеосенсора (vision_sensor). Так, первый такой геном-долгожитель появился уже во 2 поколении (Рис. 5), тогда как у сенсора близкого расстояния (proximity_sensor) - только в 39 поколении (Рис. 6).

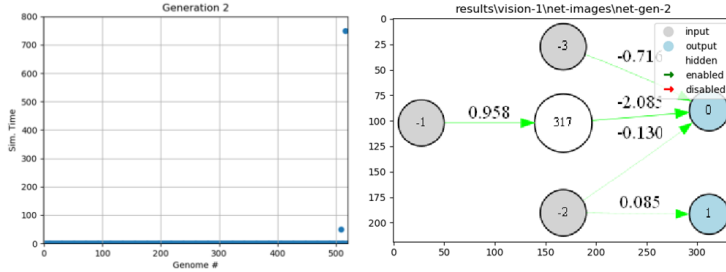


Рис. 5. Статистика генерации геномов в поколении (слева) и структура нейросети (справа) при использовании нейроэволюционного алгоритма NEAT для контроля машины в симуляторе V-REP для входных данных от видеосенсора

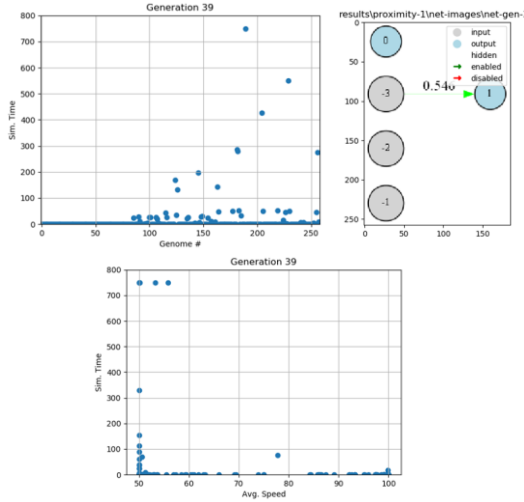


Рис. 6. Статистика генерации геномов в поколении (слева), структура нейросети (в середине) и распределение средних скоростей машины (справа) при применении нейроэволюционного алгоритма NEAT для контроля машины в среде V-REP для входных данных от сенсора расстояния

Далее поколения выбирались случайным образом из уже получившихся удачных геномов. Потом для видеосенсора на 24 поколении, а у сенсора расстояния на 42 поколении изменили сцену в симуляторе V-REP одну на другую, с более сложной трассой, записанной в файл «ackerman_car_complicated.ttt». Рассмотрим изменения результатов для видеосенсора. Сперва статистика генерации геномов в поколении испортилась (см. переход от 23-ого к 24-му поколению на Рис. 7, слева и в

середине), что не удивительно, поскольку существовавшая на тот момент нейросеть не была готова к более крутым поворотам трассы. Однако, к 39 поколению ситуация стабилизировалась и количество генов-долгожителей снова выросло (см. Рис. 7, справа). При этом структура самой сети заметно упростилась: как видно из Рис. 8, теперь для управления используется только один сенсор.

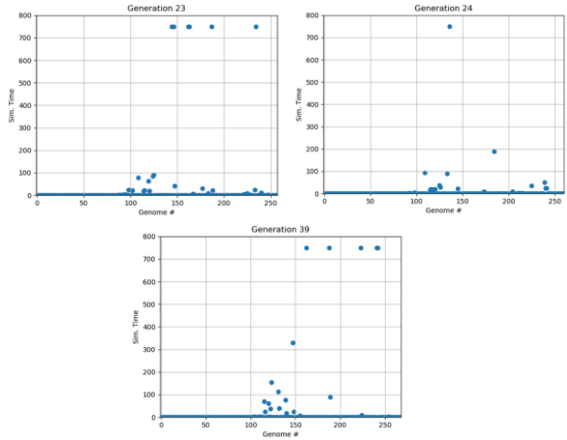


Рис. 7. Статистика генерации геномов в поколениях: 23 (слева), 24 (в середине) и 39 (справа) при использовании нейроэволюционного алгоритма NEAT для контроля машины в симуляторе V-REP для входных данных от видеосенсора

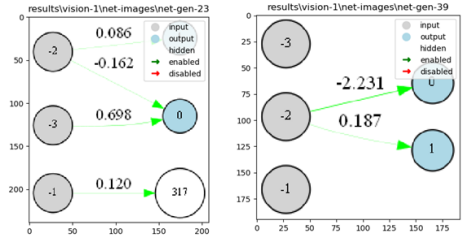


Рис. 8. Структура нейросети при генерации геномов в поколениях: 23 (слева) и 39 (справа) при использовании нейроэволюционного алгоритма NEAT для контроля машины в симуляторе V-REP для входных данных от видеосенсора

Теперь рассмотрим данные для сенсора близкого расстояния. Замена сцены на более сложную была проведена в 42 поколении, и также статистика генерации геномов испортилась при переходе от 41-ого к 42-му поколению (Рис. 9). Однако, к 56-му поколению ситуация улучшилась и генов-долгожителей стало больше (Рис. 9, справа) и структура сети стала сложнее (Рис. 10). При этом распределение средних скоростей машины для

геномов разных поколений не сильно изменилось (Рис. 6, справа и Рис. 10, справа), показывая, что более устойчивое движение машины для генов-долгожителей происходит при скорости ~50% от максимальной, что соответствует медленному движению по трассе.

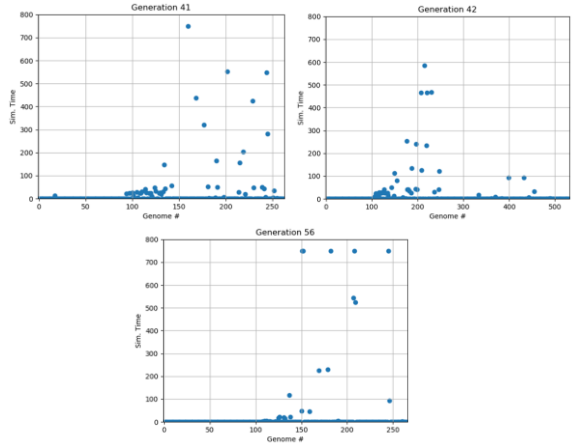


Рис. 9. Статистика генерации геномов в поколениях: 41 (слева), 42 (в середине) и 56 (справа) при контроле машины на базе NEAT по сенсору расстояния в V-REP

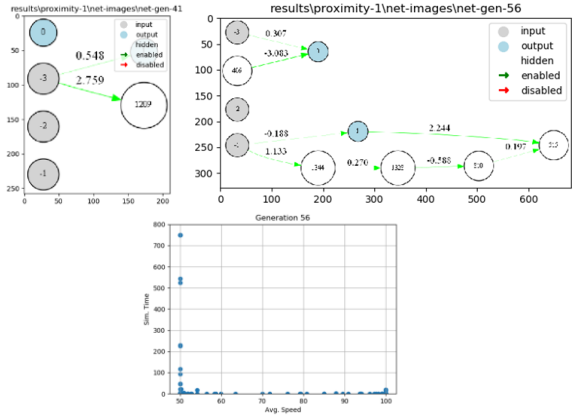


Рис. 10. Структура нейросети при генерации геномов в поколениях: 41 (слева) и 56 (в центре), а также распределение скоростей машины для 56-го поколения (справа) при контроле машины на базе NEAT по сенсору расстояния в V-REP

Заключение

В статье предложен нейроэволюционный метод контроля машины с рулевым управлением по схеме Аккермана. Для моделирования эксперимента в симуляторе V-REP был создан сценарий с трехмерной сценой и моделью машины (с соответствующими шасси, приводами, рулевым управлением и сенсорами), движение которой полностью отражает кинематические особенности аккермановской машины. Для автоматического тестирования были созданы программные модули управления машины в среде Python Remote-API и контроля схода машины с трассы. В начале моделирования, для отработки поведения машины в симуляции, использовался ПИД-контроллер, стабилизирующий движение машины вдоль линии с помощью видеосенсоров. Затем для контроля и стабилизации машины на трассе был применен нейроэволюционный алгоритм NEAT с входными данными двух типов: от видеосенсоров и от сенсоров расстояния. Нейроэволюционный контроль машины в симуляторе V-REP был исследован сперва на одной трассе, а потом валидированы на другой, показав, что нейроэволюционный подход успешно управляет аккермановской машиной, формируя через несколько десятков поколений геномы-долгожители с оптимальными параметрами контроля (как по углам рулевого управления, так и по скоростям машины) и оптимальную структуру нейросети для управления машиной.

Список литературы

[Афанасьев и др., 2015] Афанасьев и др. Навигация гетерогенной группы роботов (БПЛА и БНР) через лабиринт в 3D симуляторе Gazebo методом вероятностной дорожной карты. Сб. тр. БТС-ИИ, 18-25, 2015.

[Зенкевич и др., 2017] С.Л. Зенкевич, Ч. Хуа, Х. Цзяньвень. Движение группы мобильных роботов в строю типа “конвой” - теория, моделирование и эксперимент. Сб. трудов БТС-ИИ, 136-147, 2017.

[Лавренов и др., 2016] Лавренов Р. О., Афанасьев И.М, Магид Е.А. Планирование маршрута для беспилотного наземного робота с учетом множества критериев оптимизации. Сб. трудов БТС-ИИ, 10-20, 2016.

[Лавренов и др., 2019] Р.О. Лавренов, Е.А. Магид, Ф. Мацуно, и др. Разработка и имплементация сплайн-алгоритма планирования пути в среде ROS/Gazebo. Труды СПИИРАН, 18(1), 57-84, 2019.

[Afanasyev et al., 2015] I. Afanasyev, A. Sagitov, and E. Magid, ROS-based SLAM for a Gazebo-simulated mobile robot in image-based 3D model of indoor environment. In ACIVS. Springer, 273-283, 2015.

[Ahmadzadeh et al., 2015] H. Ahmadzadeh, E. Masehian. Modular robotic systems: Methods and algorithms for abstraction, planning, control, and synchronization. Artificial Intelligence, 223, 27-64, 2015.

[Bokovoy et al., 2018] A. Bokovoy, M. Fomin, K. Yakovlev. Implementation of the Pathfinding System for Autonomous Navigation of Mobile Ground Robot. In ITTMM-WSS, 72-78, 2018.

[Danilov et al., 2016] I. Danilov, B. Gabbasov, I. Afanasyev, and E. Magid. ZMP Trajectory from Human Body Locomotion Dynamics Evaluated by Kinect-based Motion Capture System. In VISAPP, 162-168, 2016.

[Dobriborsci et al., 2016] D. Dobriborsci, A. Kapitonov, N. Nikolaev. The basics of the identification, localization and navigation for mobile robots. In Int. Conf. on Information & Digital Technologies (IDT), 100-105, 2017.

[Filipenko et al., 2018] Filipenko M., Afanasyev I. Comparison of various slam systems for mobile robot in an indoor environment. In IEEE IS, 2018.

[Gabbasov et al., 2015] B. Gabbasov, I. Danilov, I. Afanasyev, and E. Magid. Toward a human-like biped robot gait: Biomechanical analysis of human locomotion recorded by Kinect-based Motion Capture system, ISMA, 2015.

[Ibragimov et al., 2017] Ibragimov I. et. al. Comparison of ROS-based visual slam methods in homogeneous indoor environment. In WPNC, 2017.

[Khusainov et al., 2016] Khusainov R., Shimchik I., Afanasyev I. and Magid E. 3D modelling of biped robot locomotion with walking primitives approach in Simulink environment. LNEE (383): 287-304. Springer, 2016.

[Lavrenov et al., 2017] Lavrenov R., Zakiev A. Tool for 3D Gazebo Map Construction from Arbitrary Images and Laser Scans. In DeSE, 2017.

[Magid et al., 2017] Magid E., Lavrenov R., Afanasyev I. Voronoi-based trajectory optimization for UGV path planning. In ICMSC, 383-387, 2017.

[Rohmer et al., 2013] E. Rohmer, S.P. Singh, M. Freese. V-REP: A versatile and scalable robot simulation framework. In IEEE IROS, 1321-1326, 2013.

[Sheikh et al., 2018] Sheikh T.S., Afanasyev I.M. Stereo vision-based optimal path planning with stochastic maps for mobile robot navigation. Advances in Intelligent Systems and Computing, vol 867, 40-55, Springer, 2018.

[Shimchik et al., 2016] Shimchik I., Sagitov A., Afanasyev I., Matsuno F., Magid E. Golf cart prototype development and navigation simulation using ROS and Gazebo. In MATEC Web of Conferences, vol. 75, p. 09005, 2016.

[Sokolov et al., 2016] Sokolov M. et al. 3D modelling and simulation of a crawler robot in ROS/Gazebo. In ICMA, ACM, 61-65, 2016.

[Sokolov et al., 2017a] Sokolov M. et al. Modelling a crawler-type UGV for urban search and rescue in Gazebo environment. In Proc. ICAROB, 2017.

[Sokolov et al., 2017b] Sokolov M. et al. HyperNEAT-based flipper control for a crawler robot motion in 3D simulation environment. In ROBIO, 2017.

[Stanley et al, 2002] K.O. Stanley, R. Miikkulainen. Efficient evolution of neural network topologies. In IEEE CEC, vol. 2, 1757-1762, 2002.

[Zubov et al, 2018] Zubov I., Afanasyev I., Shimchik I., Mustafin R., Gabdullin A. Autonomous Drifting Control in 3D Car Racing Simulator. In IEEE Intelligent Systems (IS), 2018.

УДК 004.896

СИСТЕМА УПРАВЛЕНИЯ ДВИЖЕНИЯМИ АНТРОПОМОРФНОГО РОБОТА-ВОДИТЕЛЯ НА ОСНОВЕ ФОРМАЛЬНОЙ ГРАММАТИКИ

П.С. Сорокоумов (*petr.sorokoumov@gmail.com*)
НИЦ «Курчатовский институт», Москва

Аннотация. Разработка роботов, способных действовать совместно с людьми в одной среде, является весьма важной и актуальной задачей. Помимо прочего такие роботы должны работать с созданными для человека средствами ручного управления: кнопками, рукоятками, рулевыми колёсами и т.п. Сейчас манипуляции с ними для роботов весьма сложны, потому что контроль движений конечностей и органы захвата устроены у них иначе, чем у людей, поэтому большую важность приобретают средства, которые позволяют адекватно описать такие задачи и применять полученные описания для разработки эффективных систем управления движениями. В данной работе предлагается подход к описанию таких манипуляций с помощью формальной грамматики. Её нетерминальными символами являются двигательные задачи для конечностей робота, а терминальными - их геометрически примитивные участки. Таким образом, грамматика описывает специфику работы со средством управления без привязки к конкретной модели робота. Составленное описание позволяет разделить задачу на последовательные компоненты, которые позволяют воспроизвести нужную траекторию. Движение при этом не сводится к копированию заложенных образцов, а корректируется на каждом этапе для эффективного достижения поставленной цели. Благодаря тому, что количество разных типов требуемых для манипулирования движений невелико, описание носит достаточно универсальный характер. Предложенный метод была использован для моделирования движений антропоморфного робота-водителя, потенциально способного работать со средствами управления обычного, не подготавливаемого специально автомобиля[1].

Ключевые слова: робот-водитель, управление манипулятором, формальная грамматика.

[1] Работа выполнена при поддержке НИЦ «Курчатовский институт» (приказ №1601 от 05.07.2018г.)

1 Введение

В лаборатории робототехники НБИКС ПТ НИЦ «Курчатовский институт» в настоящее время ведётся работа по созданию робота, способного управлять автомобилем. Основными отличиями целей данного проекта от других многочисленных разрабатываемых систем роботизации автотранспорта являются:

- необходимость работы в немодифицированном автомобиле: робот должен располагаться на стандартном рабочем месте водителя-человека без существенных технических модификаций;
- требования к универсальности полученных решений: как математические, так и технические компоненты систем должны не просто обеспечивать работу водителя в её узком понимании, но обобщаться на более широкий класс задач – взаимодействие роботов с окружением, созданным для людей, и успешное их функционирование в этом окружении.

Основной акцент при разработке делается на ручном управлении автомобилем под руководством внешней системы, самостоятельно решающей задачи прокладки маршрута, анализа дорожной обстановки и выбора необходимых действий водителя.

В ручном управлении автомобилем с помощью робота можно выделить несколько подзадач. Сначала робот, помещённый в кабину, распознаёт элементы управления автомобилем по сенсорным данным и определяет их положение в пространстве. Далее он должен манипулировать органами управления, решая поставленные задачи. В процессе работы возможна потеря сенсорного контакта с органами управления, требующая повторного (частичного) распознавания окружения. Из трёх перечисленных подзадач (первоначальное ориентирование, выполнение манипуляций, дополнительное ориентирование) решение первой и третьей для нескольких характерных типов органов управления было описано ранее [Сорокоумов, 2017]. В контексте данной работы они считаются решёнными с приемлемой точностью.

Современным роботам пока сложно манипулировать органами управления, созданными для человека. В работе водителя имеются некоторые дополнительные факторы, облегчающие роботизацию: робот не должен поддерживать равновесие, потому что он при вождении сидит; количество типов органов управления невелико; ожидаемые реакции от них стандартны. Могут усложнить манипулирование внешние воздействия (тряска, инерциальные эффекты).

Для тестирования разработок используется стенд, состоящий из антропоморфного робота REEM-C и макета рабочего места водителя (рис. 1).

Рис. 1. Общий вид макета рабочего места водителя с антропоморфным роботом REEM-C

2 Материалы и методы

2.1 Специфика задачи манипулирования органами ручного управления

Для перемещения органов ручного управления манипуляторами робота по заданной траектории можно использовать давно и хорошо известные методы решения задачи обратной кинематики, входящие в состав стандартных программных пакетов для работы с роботами, например, MoveIt в ROS. Однако поставленную задачу надо сначала свести к задаче обратной кинематики. Это преобразование осложняется множеством факторов:

- некоторые движения можно выполнить только одной конечностью (переключение передач выполняется правой рукой), другие – несколькими разными, некоторые – несколькими, действующими одновременно (движение рулевого колеса можно выполнять двумя руками, можно – одной);
- необходимо обеспечивать своевременный захват рукояток и их отпускание, причём для разных органов управления тип захвата будет различаться;

- движения должны иногда выполняться одновременно; часто при этом требуется синхронизация.

Исходя из изложенных факторов, понятно, что нахождение общей концептуальной основы для представления движений, позволяющей вырабатывать решение с учётом этих факторов, весьма интересно.

2.2 Формальная грамматика как форма описания элементов управления

Целесообразным представляется найти описание органов ручного управления техническими системами, которое достаточно хорошо подходило бы для работы с ними автоматических систем манипулирования. Так как траектории движения рук при действиях достаточно просты, то можно попытаться описать их в виде набора геометрических примитивов.

Давно известно, что описание контура можно выполнить в виде формального языка, имеющего в качестве терминалов примитивные геометрические элементы контуров [Fu, 1974]. Если описать движения какого-либо органа управления таким образом и добавить терминальные символы, отвечающие за прочие действия с органом управления (захват, перехват, отпускание), то можно получить единый способ представления действий, пригодный для их трансляции в движения робота. Решения подобного рода предлагались неоднократно: например, в [Bidgoli et al, 2014] предлагается использование такого представления в манипуляторе-раскройщике материала. Для рассматриваемой задачи этот метод имеет важное преимущество: так как количество разных типов органов управления невелико, то можно построить описание небольшого размера, способное решить широкий спектр задач.

Общая архитектура предлагаемого решения представлена на рис. 2. От блока постановки задач на синтаксическую обработку поступают команды, воспринимаемые как нетерминальные символы некоторой грамматики, для которых требуется получить представление в виде последовательности терминалов. Полученные терминалы интерпретируются как команды манипуляторам робота и исполняются. При этом исполнение операций должно корректироваться сенсорными данными прозрачно для синтаксического блока.

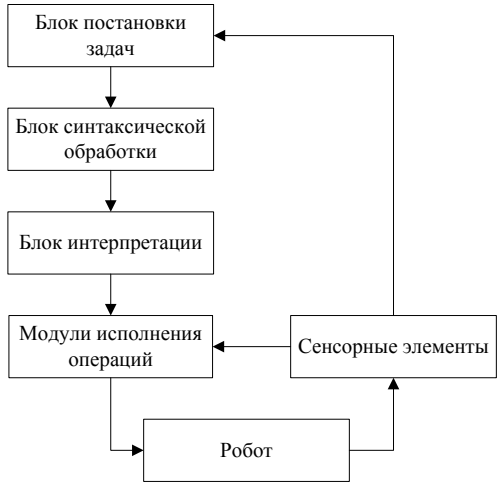


Рис. 2. Общая архитектура системы управления движением робота

2.3 Пример описания органа управления

Для описания работы с некоторыми типичными органами управления автомобилем можно использовать наборы терминальных символов (элементарных движений), показанные на рис. 3. Стрелке соответствует один терминальный символ, точке – два: захват, интерпретируемый как создание устойчивого контакта манипулятора и органа, и отпускание, интерпретируемое как отстранение манипулятора с разрушением контакта. Имена соответствующих точкам терминальных символов получаются из их названий приписыванием обозначений G (grab) или R (release).

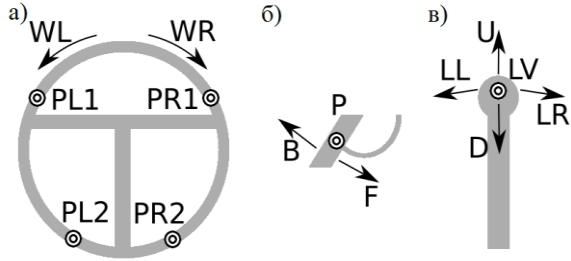


Рис. 3. Символы, требуемые для описания некоторых элементов ручного управления: а – рулевого колеса, б – педали, в – рычага.

Например, рулевое колесо (рис. 3а) управляется путём захвата либо отпускания точек из выделенного набора, включающего в примере две

точки для каждой руки P(L|R)(1|2). Поворот руля осуществляется примитивами движения конечности по дуге на единое жёстко заданное значение WR и WL. Таким образом, набор терминальных символов для рулевого колеса состоит из 10 элементов (два движения WL и WR, выполняемые каждой из двух рук, а также захват и отпускание каждой из четырёх точек). В роли команд выступают четыре нетерминала: ToLeft и ToRight, соответствующие повороту руля направо и налево на заданный угол, Break, прекращающий поворот руля, и Finish, завершающий работу робота с рулём. В таком случае простейший вариант контекстно-свободной грамматики, позволяющей описывать манипуляции с рулём, имеет следующий вид (терминальные символы выделены полужирным шрифтом):

S → Finish | GrabL ManipulateL ReleaseL S | Grab Manipulate Release S
ManipulateL → MoveL | MoveL ManipulateL | MoveL ReleaseL GrabL ManipulateL
Manipulate → Move | Move Manipulate | Move Release Grab Manipulate
Move → Break | ToLeft **LWL RWL** Move | ToRight **LWR RWR** Move
MoveL → Break | ToLeft **LWL** MoveL | ToRight **LWR** MoveL
Grab → GrabL GrabR
GrabL → **PL1** | **PL2**
GrabR → **PR1** | **PR2**
Release → ReleaseL ReleaseR
ReleaseL → **LR** # для отпускания руля каждой рукой
ReleaseR → **RR** # требуется одна команда
Break → **BRK**
ToLeft → ø
ToRight → ø
Finish → **FIN**

В показанной версии грамматики учтено, что водитель может поворачивать руль как двумя руками, так и одной левой рукой. Команды при этом порождаются символами Manipulate или ManipulateL соответственно. Терминальные команды движения, предназначенные для правой либо левой руки, различаются первой буквой названия – R или L.

Данная грамматика позволяет получить любую корректную последовательность действий с рулевым колесом. При анализе полученных из центра управления нетерминальных символов модуль синтаксического анализа выбирает, какая строка терминалов должна быть порождена. Контекст исполнения при этом обрабатывается модулем исполнения операций, который по сведениям о геометрии системы интерпретирует команду движения руки робота по окружности обода рулевого колеса в

траекторию, начинающуюся от текущего положения руки на колесе. Поскольку выбранные команды-нетерминалы не позволяют указать, как конкретно робот должен захватывать руль: одной рукой или двумя, в каких именно точках – то блок синтаксической обработки может самостоятельно выбирать любой подходящий вариант захвата. Аналогичным образом обеспечивается и перехват рук на руле.

2.4 Алгоритм порождения управляющей последовательности

Последовательность терминальных символов порождается из последовательности команд-нетерминалов, присылаемых блоком постановки задач, общеизвестными методами генерации текста по описывающей его контекстно-свободной грамматике.

Состояние S процесса порождения можно описать множеством троек (T, C, N). Каждая тройка задаёт одну из возможных реакций системы на команду C как последовательность порождаемых терминальных символов T и множество последовательностей символов N, задающих возможные следующие состояния процесса. Для наглядности рассмотрим одну из таких троек, формируемую по описанной выше грамматике сразу после инициализации процесса, до приёма команд:

(T = [**PL1**], C = ToLeft,
N = [[**LWL**, MoveL, ManipulateL, ReleaseL, S],
[**LWL**, MoveL, ReleaseL, GrabL, ManipulateL, ReleaseL, S],
[**LWL**, MoveL, ReleaseL, S]).

Данная тройка показывает, что одной из возможных реакций системы на команду ToLeft является отсылка блоку интерпретации инструкции PL1 (команды захвата руля левой рукой), причём после выбора этого варианта действий дальнейшая работа может пойти по трём путям. После единичного движения руля налево одной левой рукой командой LWL и дальнейших движений влево, порождаемых MoveL, возможны выполнение дальнейших манипуляций левой рукой (первый член N), перехват руля левой рукой и дальнейшие действия ею (второй член N) либо прекращение манипуляций с возможностью их возобновления (третий член N).

Процесс порождения реализован в виде двух повторяющихся шагов – выбора реакции на команду и обновления состояния. После получения команды необходимо выбрать одну из троек, реагирующих на неё. В данной реализации выбор осуществляется случайным образом, так как любой предлагаемый вариант должен работать корректно; в общем случае можно выбирать наилучший вариант на основе сенсорных данных о состоянии процесса. Если нужных троек нет, команда игнорируется как недопустимая в текущем состоянии. Далее терминальные символы выбранной тройки T передаются для исполнения интерпретатору, а из N порождается новое состояние процесса.

Процесс порождения состоит в том, что в каждом варианте следующего состояния выполняются все порождения самого левого нетерминала до тех пор, пока этим самым левым нетерминалом не окажется команда; после этого из результата формируется тройка показанного выше вида. Такое обновление состояния завершится за конечное число шагов, только если грамматика не способна породить бесконечное множество идущих подряд терминальных символов без использования нетерминалов из набора команд. Это - основное требование к используемой грамматике.

2.5 Требования к интерпретатору и исполнительным модулям

Управление действиями робота ведёт интерпретатор, контролирующий ход элементарных двигательных операций, проводимых исполнительными модулями. В разрабатываемой системе управления к этим компонентам предъявляются следующие требования:

- одновременно полученные команды, относящиеся к разным конечностям, исполняются одновременно;
- при одновременном появлении команд, влияющих на один орган управления (LWL и RWL в примере выше), процессы их исполнения становятся взаимозависимыми (синхронизируются);
- модули исполнения операций обеспечивают плавность движений.

Проще всего удовлетворить эти требования, заставив интерпретатор работать синхронно с блоком постановки задач и блоком синтаксического анализа. Длительность такта при этом определяется длительностью самого долгого из исполняемых двигательных действий, поэтому продолжительности разных тактов могут различаться.

В настоящий момент ведётся разработка интерпретатора и модулей исполнения операций, способных удовлетворить изложенные требования для работы системы с тестовым роботом Reem-C.

2.6 Параллельные грамматики для обеспечения синхронии движений

Для совместной работы с множеством разных органов управления, каждый из которых представлен собственной грамматикой, необходимо использовать общий метод синхронизации. Одним из подходов к этой задаче является объединение отдельных грамматик в единую параллельную грамматическую систему с коммуникацией (parallel communicating grammar system). Эта конструкция представляет собой набор грамматик, в которых помимо обычных терминальных и нетерминальных символов могут

появляться символы запроса, т.е. обращения к правилам другой заданной грамматики системы [Csuhaj-Varju et al, 1990].

Среди параллельных грамматик выделяют синхронные и асинхронные. Синхронные грамматики более выразительны, но из-за сложности их задания и проверки корректности в данной работе использован асинхронный вариант. Для создания параллельной грамматики в этом случае достаточно механически объединить компоненты, соответствующие разным элементам управления, при необходимости переименовав совпадающие термы разных грамматик во избежание коллизий. Синхронизация в такой системе обеспечивается введением дополнительной центральной грамматики, которая занимается главным образом передачей команд другим грамматикам в правильной последовательности [Paun, 1993]. На практике для синхронизации может также потребоваться введение дополнительных термов в существующие грамматики. Например, чтобы запретить захват руля двумя руками при необходимости работать с рычагом, можно ввести в первое правило грамматики из примера нетерминалы, отвечающие за работу с рычагом:

S → Finish | WorkWithLever GrabL ManipulateL StopWorkWithLever ReleaseL S | Grab Manipulate Release S

После этого использование одноручного захвата руля становится возможным только после прихода команд манипуляций с рычагом, иначе обращения к соответствущим правилам не происходит. В настоящий момент ведутся дополнительные исследования методов, позволяющих избежать модификации исходных грамматик.

3 Реализация и обсуждение

Предложенная система была реализована в виде компонента синтеза управляющих последовательностей по параллельным грамматикам и набора процедур исполнения элементарных движений. Процедуры исполнения построены в виде последовательностей обращений к библиотеке алгоритмов обратной задачи кинематики MoveIt, входящей в состав фреймворка ROS. При работе с грамматиками использованы средства платформы NLTK. Полученные наборы команд анализировались на корректность на упрощённой программной модели задачи, оценивающей правильность . Алгоритм смог обработать тестовые последовательности высокоуровневых команд, имитирующих простейшие действия водителя (трогание с места и перестроение). Длительность одного цикла синтаксической обработки на тестовой последовательности не превышала 10 мс, что приемлемо для управления целевым роботом.

Результаты моделирования говорят о том, что выбранный подход для высокоуровневого описания органов управления позволяет получить

достаточно качественное решение, позволяющее организовать работу робота-водителя. Однако для практического применения данной разработки требуется детальное исследование качества управления на реальном роботе.

Наиболее значительным недостатком предложенной системы является сложность синхронизации конечностей в процессе выполнения элементарных действий, так как для устойчивой работы, например, двух рук при совместном повороте руля синхронизации при запуске действий недостаточно. Значительной проблемой является придание последовательности выбранных действий временной составляющей при синтаксической обработке. В качестве направления будущих исследований можно также наметить получение необходимого набора геометрических примитивов непосредственно из данных о движениях реальных операторов и создание грамматик на их основе.

Предложенное решение носит достаточно универсальный характер и может использоваться в общих задачах взаимодействия манипулятора с внешней средой.

Благодарности. Автор считает долгом поблагодарить всех сотрудников лаборатории робототехники КК НБИКС ПТ и в особенности Е.П. Орлинского и Д.М. Павлова, без упорного труда которых данная разработка была бы невозможна.

Список литературы

[**Bidgoli et al, 2014]** Bidgoli A., Cardoso-llach D. Towards a motion grammar for robotic stereotomy // 2014. № Daas.

[Csuhaj-Varju et al, 1990] Csuhaj-Varju E., Dassow J. On Cooperating/Distributed Grammar Systems // J. Inf. Process. Cybern. 1990. Т. 26. № 1–2. с. 49–63.

[Fu, 1974] Fu K.S. Syntactic Methods in Pattern Recognition. New York: Academic Press, 1974.

[Paun, 1993] Paun G. On the synchronization in parallel communicating grammar systems // Acta Inform. 1993. Т. 30. с. 351–367.

[Сорокоумов, 2017] Сорокоумов П.С. Система компьютерного зрения для распознавания элементов управления автомобилем роботом-водителем // IV Всероссийский научно-практический семинар «Беспилотные транспортные средства с элементами искусственного интеллекта» (БТС-ИИ-2017), Казань, 5-6 октября 2017. Казань: Центр инновационных технологий, 2017. с. 46–55.

УДК 681.5+004.8

МОБИЛЬНЫЙ СОЦИАЛЬНЫЙ РОБОТ

Н.В. Ким (*nkim2011@ list.ru*)
Московский авиационный институт, Москва

В.Н. Жидков (*vladimir_zhidkov@mail.ru*)
Московский авиационный институт, Москва

В.Н. Пименов (*vladimir.pimenov@oaoapz.com*)
Арзамасский приборостроительный завод, Арзамас

Аннотация. В статье обсуждаются вопросы разработки социального мобильного робота, обеспечивающего поддержку больным, людям с ограниченными возможностями, одиноким пожилым людям. Робот предназначен для передачи лекарств, напитков, небольших предметов ограниченным в перемещении пользователям. Поведение робота основано на выполнении ряда базовых сценариев и механизме адаптации к конкретному пользователю. Приведены примеры реализации отдельных этапов выполнения сценариев.

Ключевые слова: социальный мобильный робот, сценарии поведения, навигация робота, планирование, адаптация, БТС-ИИ.

Введение

В последние годы широкое распространение в ряде стран получила социальная робототехника. Социальные роботы (СР) обеспечивают психологическую, информационную и физическую поддержку больным, людям с ограниченными возможностями, одиноким пожилым людям [1, 2].

К данной категории роботов относят, в частности, мобильных роботов андроидного типа [3, 4, 5] которые могут двигаться, с помощью манипуляторов брать и передавать различные лекарства, напитки, небольшие предметы ограниченным в перемещении пользователям:

Promobot («Промобот», Россия) [6] – автономный робот, способный общаться с людьми, распознавать лица, отвечать на вопросы, перемещаться, избегая столкновений, двигать руками и головой, транслировать различные материалы на своём дисплее.

Asimo (Honda, Япония) [7] умеет взаимодействовать с людьми, следовать за ними, избегая столкновений, может переносить различные предметы, например, поднос с напитками.

Nexi (MIT, США) понимает эмоции собеседника и имитирует свои. Передвижение робота обеспечивается посредством мобильной базы на двух колёсах. Руки робота выдерживают груз до 3,5 кг. Этот социальный робот успешно тестировался в нескольких американских домах престарелых.

RIBA-II (RIKEN, Япония) [8, 9] может поднять пациента с пола, перенести пациента с кровати в коляску и обратно.

На основании анализа функциональных возможностей известных социальных роботов авторами была сформирована концепция разработки мобильного социального помощника пользователя.

1 Архитектура и алгоритмы

Данный социальный робот (СР) оснащен системой технического зрения (СТЗ), транспортной (колесной) платформой (ТП), двумя манипуляторами (руками) и предназначен для доставки и передачи пользователю/пациенту (П) необходимых объектов, например, лекарств, напитков и пр. Проектные характеристики СР: высота – 1200 мм, вес – 12 кГ, длина манипулятора – 600 мм, номинальная скорость – 30 мм/с.

Поведение СР основано на выполнении базовых сценариев. Разработано 12 сценариев, определяющих поведение СР в случае различных обращений, воздействий на СР пользователя, при необходимости ответов на запросы, выполнении задаваемых функций и пр. Определены этапы выполнения сценариев.

Например, *Сценарий 1* определяет действия при передаче пользователю объектов (O_i), не размещенных на СР, где i – индекс объекта; *Сценарий 2* – организация связи с внешними абонентами; *Сценарий 5* – организация диалогов; *Сценарий 7* – диагностика состояния пациента и т.д.

Сценарий 1 реализуется *по речевому запросу* П, например, «Принеси воды» или «Дай лекарство» и т.д. или *без запроса,* например, при наступлении времени принятия лекарств.

Выполнение данного сценария производится по этапам:

1. Оценка собственного положения ТП,
2. Поиск O_i,
3. Планирование маршрута к O_i,
4. Перемещение к O_i и захват O_i манипулятором,
5. Поиск и оценка положения П,
6. Планирование маршрута к П,
7. Перемещение СР к пользователю в область допустимого контакта,
8. Передача O_i пользователю.

Если требуется передача пользователю O_j, размещенных на СР, то из *Сценария 1* исключаются этапы 2, 3, 4.

Выполнение этапа 1 (оценка собственного положения ТП) реализуется с использованием внешних, по отношению к ТП средств измерения. Измерительная система состоит из нескольких ультразвуковых приемников, установленных по периметру рабочего пространства и ультразвукового излучателя, установленного на ТП. Работа излучателя и приемников синхронизирована во времени. Измеряются задержки времени между моментом излучения сигнала излучателя, на основании которых определяются дальности до соответствующих приемников.

На основании измеренных дальностей, при известных координатах приемников излучения оцениваются координаты ТП с помощью обобщенного фильтра Калмана.

Поиск O_i и П (этапы 2, 5) производится СТЗ в рамках технологии сплошного поиска. Сложность решения данной задачи связана с необходимостью поиска в условиях изменяемой освещенности, неопределенности положения O_i и П и состава наблюдаемых сцен.

Планирование маршрутов (этапы 3, 6) и управление перемещением ТП (этапы 4, 7) реализованы известными методами с использованием волнового алгоритма планирования.

Реализация указанных процедур на разрабатываемом роботе-демонстраторе показало недопустимо высокую загрузку бортовых вычислителей СР при его работе в реальном времени.

Сокращение вычислений, необходимых для решения поставленных задач, в частности, планирования, оптимизация траекторий движения ТП могут быть реализованы за счет классификации возникающих ситуаций.

При этом, если однотипные действия робота повторяются регулярно можно говорить об обычном поведении. Обслуживание в рамках обычного поведения пользователя имеет ряд преимуществ:

- возможность решения навигационных задач в локальных заранее определенных областях;

-переход к алгоритмам вторичного поиска, более компактным, чем сплошной поиск;

- использование базовых траекторий движения с локальными коррекциями маршрута;

- оптимизация движений манипуляторов и схватов;

- сокращение словаря признаков и рабочего алфавита классов объектов интереса и пр.

В любом случае робот должен уметь отслеживать повторяющиеся просьбы и задания, и находить причинно-следственные связи между просьбами и заданиями, получаемыми от пользователя и возможными причинами этих заданий, т.е. привязывать их ко времени дня, дню недели, перечню предшествующих событий. Для нахождения причинно-следственных связей могут использоваться хорошо известные методы и

алгоритмы, такие как ID3, ДСМ-метод и другие. При невозможности автоматического определения причинно-следственных связей, например, потому что в базе знаний робота отсутствует информация о факторе, который оказался определяющим, вполне допустим вопрос к пользователю о причинах полученного задания и введение нового фактора в базу знаний робота.

Классификация ситуаций и выполняемых операций позволяет оптимизировать планирование действий и сформировать «характер» робота, ориентированный на обслуживание конкретного пользователя.

2 Адаптация к пользователю

Позволяет оптимизировать траектории движения робота, количество перемещений, время общения (количество задаваемых вопросов), повысить дружественность общения, услужливость, тактичность робота.

Адаптация может быть реализована как с использованием механизмов самообучения, так и обучения с учителем. Если поведение робота не нравится человеку, то робот фиксирует цепочку принятия решения и значения сопутствующих факторов, которая привела к выполнению роботом действий, не понравившимся человеку. Эта цепочка принятия решения может быть использована как пример для обучения системы выбора сценария поведения робота. При обучении "без учителя" человек непосредственно не принимает участия в процессе обучения, и робот самостоятельно дает оценку выбранному сценарию поведения, основываясь на зафиксированной реакции человека. Система выбора сценария поведения робота может быть реализована как экспертная система основанная на правилах (в том числе и нечетких), на основе искусственных нейронных сетей, или как "нейро-нечеткая". Схема обучения "без учителя" представлена на Рис.1.

Система контроля эмоций клиента - автоматическая система, которая на основании анализа изображения лица клиента, анализа голоса клиента оценивает степень удовлетворенности клиента действиями робота.

Алгоритм принятия решения определяет сценарий поведения робота.

Блок "формализация окружающей среды" создает формальное описание окружающей среды на языке, понятном блоку принятия решения.

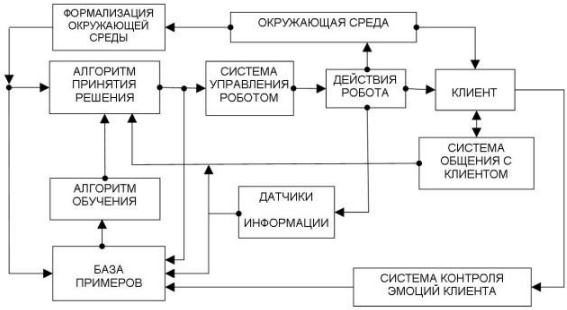


Рис. 1. Обучение социального робота "без учителя"

При обучении "с учителем" (Рис.2) в роли учителя выступает человек - клиент, с которым робот общается. Получив информацию, что его действия не нравятся клиенту, робот должен спросить его о том, почему выбранный сценарий поведения вызвал негативную реакцию. Если клиент называет соответствующие факторы, то они вместе с описанием текущей ситуации и предшествующими этой ситуации событиями, также используются как пример для обучения системы выбора сценария поведения робота.

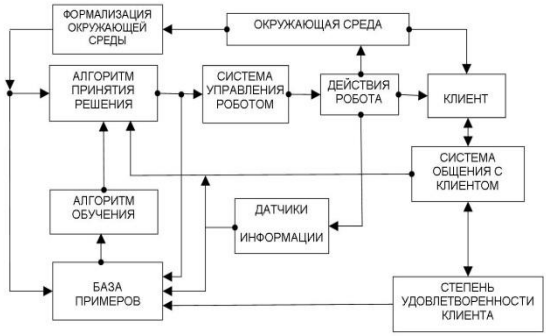


Рис.2. Обучение социального робота "с учителем"

В роли учителя также может выступать внешний эксперт, который имеет возможность дистанционно оценивать действия робота в той или иной ситуации и вносить изменения в базу знаний управляющей экспертной системы.

Заключение

На основе анализа характеристик известных мобильных роботов определена структура социального робота, предназначенного для обеспечения помощи ограниченным в перемещении пользователям. Разработаны базовые сценарии поведения роботов. Показано, что для эффективной работы необходимо классифицировать поведение робота с учетом предпочтений пользователя. Подобная классификация позволит сформировать «характер» робота, ориентированный на обслуживание конкретного пользователя.

Список литературы

1. http://topor.info/tops/socialnyjj-robot
2. https://robotics.ua
3. http://jujo.ru/style/gaget/8477-jarpet_virtualnii_pitomec_v_banke__innovacionnii_tamagochi_dlya_sovremennogo_rebenka.html ,
4. http://www.pvsm.ru/news/93221
5. http://www.rusmedserver.ru/med/bolezni/articls/20.html
6. https://tymolod59.ru/2999
7. https://masterok.livejournal.com/4573997.html
8. http://roboting.ru/1448-robot-sidelka-riba-ii.html
9. http://miuki.info//12/yaponskie-roboty-video-foto/

УДК 004.021, 007.52

АЛГОРИТМ ДИНАМИЧЕСКОГО ФОРМИРОВАНИЯ СТАИ

В.В. Воробьев (*gatus86@mail.ru*)
НИЦ “Курчатовский институт”, Москва

Аннотация. В работе рассматривается решение задачи выбора лидера в группе роботов. Особенностью предлагаемого решения является возможность определять лидера таким образом, чтобы он располагался близко к топологическому центру динамически формируемой группы роботов. При решении последующих задач, которые требуют интенсивного обмена данными между лидером и периферийными роботами, такая конфигурация обеспечивает минимизацию времени, которое для этого необходимо. В основе предлагаемого механизма лежит обмен сообщениями между каждым отдельным роботом и его локальными соседями.[1]

Ключевые слова: групповая робототехника, выбор лидера, статический рой, локальное взаимодействие.

Введение

Задача выбора лидера относится к категории фундаментальных задач групповой робототехники [Dieudonné et al., 2010], о чем также говорит и большое число работ и подходов к решению этой проблемы.

Важность задачи выбора лидера обусловлена тем, что наличие лидера в группе роботов позволяет более эффективно управлять этой группой и контролировать ее. Иерархия также необходима при решении задач планирования, контроля над исполнением плана и т.д., которые неизбежно возникают при решении сложных групповых задач, например, сохранения энергетического гомеостазиса группы, выполнения запросов оператора и т.п. При использовании механизмов социального поведения в основе системы управления роботами (см., например, [Карпов, 2016]) также, так или иначе, требуются механизмы, способные выделить доминантных особей в группе таких роботов.

Процесс выбора лидера описан применительно к сетевым структурам в [Lynch et al., 1993]. Представлены алгоритмы Ле-Ланна, его модификация –

[1] Работа выполнена при частичной финансовой поддержке РФФИ (проект № 16-29-04412 офи_м).

алгоритм Ченя-Робертса, Хиршберга-Синклера, Фредриксона-Линча и т.д. для кольцевых структур. В основном алгоритмы основаны на обмене уникальными идентификаторами. Например, в случае алгоритма Ле-Ланна каждый элемент сети отправляет свой маркер с отличительным признаком (весом). Когда маркер возвращается отправителю остальные маркеры также прошли через него, следовательно, отправитель сможет выбрать маркер с наименьшим или наибольшим весом.

Решение подобной задачи описывается в [Santoro, 2007], где представлены сразу несколько стратегий выбора лидера, которые, однако, также подходят более для вычислительных сетей, нежели для группы роботов. В работе рассматриваются различные топологии сети и алгоритмы выбора лидера для них. Кроме того, представлен и ряд “универсальных” алгоритмов, наиболее интересный из которых YO-YO. Принцип его работы заключается в обмене заранее заданными весами между вычислительными устройствами. Лидером становится устройство с наименьшим весом. При этом алгоритм обладает стадией подготовки, что не всегда удобно, особенно для групповой робототехники.

Еще одним решением является определение лидера с помощью оценки его расстояния до центра определенной окружности для группы анонимных роботов [Canepa et al., 2007]. Робот становится лидером, если это расстояние для него минимально. Если таковых нет, то с некоторой вероятностью роботы с некоторой вероятностью двигаются к центру этой окружности. Расстояние тем больше, чем они дальше от центра. Также описан случай с 3-я роботами, когда лидером становится тот, чья разница углов, вычисляемых для всех соседей минимальна. Таким образом, роботы должны иметь возможность двигаться и иметь средство связи со всеми роботами группы, например, доску объявлений.

Похожим образом задача решается в [Flocchini et al., 2008], где лидер определяется в соответствии с заданным шаблоном – тот, кто был ближе к заданной шаблоном точке, тот и становится лидером. При этом роботы анонимны и не имеют общей координатной системы.

Другой подход к выбору лидера, который разрабатывался специально для групп роботов, описан в [Gan Chaudhuri et al., 2015]. В рассматриваемой модели роботы не имеют общей координатной системы и уникальных идентификаторов, группировка гомогенна. При этом роботы не могут опираться на информацию о вычислениях и сенсорных данных, сделанных ранее. Выбор лидера происходит по принципу близости робота к центру определенной формации, т.е. роботы сначала формируют некий паттерн, который одновременно является чем-то вроде глобальной системы координат, а затем определяются финальные позиции каждого робота. В конце они стараются достичь данных позиций. Тот робот, который

окажется ближе всего к центру описанной окружности данной формации и станет лидером.

Еще один механизм выбора лидера описан в [Karpov et al., 2015], суть которого, заключается в том, что робот определяет за кого проголосовали его соседи. В зависимости от веса кандидата, за которого голосует его сосед, робот может поменять свой выбор и проголосовать за того же кандидата. Особенностью такого подхода является то, что нет необходимости в ведении уникальных идентификаторов роботов, а также то, что они используют только локальное взаимодействие друг с другом. Сам процесс выбора лидера происходит в структуре, называемой статическим роем – некой фиксированной в определенный момент времени сети, состоящая из роботов, соединенных друг с другом по каналам связи [Карпов, 2013].

Специфика рассмотренных методов заключается в том, что лидером может стать любой робот, в том числе и периферийный, т.е. тот, кто находится на краю группы. В условиях, когда роботы общаются исключительно локально, т.е. только с ограниченным числом своих соседей, может возникнуть ситуация, когда время обмена данными с лидером существенно увеличивается. Поэтому важно чтобы лидером становился робот, который находится близко к центру группы. Подобный алгоритм, который позволяет так назначать лидера, был описан в [Воробьев, 2017]. Однако его особенностью является то, что группа роботов заранее сформирована в статический рой [Карпов, 2013], а начало процедуры выбора лидера инициируется сигналом извне, например, с помощью команды оператора. В реальных задачах это не всегда возможно, другими словами специфика решения такой задачи должна учитывать не только топологические характеристики группировки, т.е. взаимное расположение роботов, но и его динамику, т.е. изменение топологии группы в процессе появления новых роботов/отключения уже включенных в группу. Сложность заключается в том, что это можно оценить только в процессе обмена информацией друг с другом по локальным каналам связи.

Таким образом, можно отметить, что некоторые алгоритмы выбора лидера ориентируются в основном на сетевые структуры, где не учитывается возможность перемещения узлов сети, что важно при использовании в робототехнике. Ряд других методов выбирает лидера по степени геометрической близости к определенной точке или с помощью обмена весами. При этом на группу роботов накладываются существенные ограничения, например, они анонимны, не имеют общей координатной системы, могут общаться только локально. Однако не рассматривается вопрос дальнейшего обмена информацией между лидером и остальными членами группы, который особенно важен в случае локальности взаимодействия роботов друг с другом. В таком случае, если

характеристики канала связи между соседями не зависят от расстояния или ими можно пренебречь, важно, чтобы лидером стал робот, близкий к топологическому центру этой группы.

1 Постановка задачи

Предположим, что есть гомогенная группировка роботов численности N, каждый из которых описывается шестеркой $V=(\alpha,L,C,W,W_{my},P)$, где $\alpha > 0$ – уникальный идентификатор робота, L – список его соседей, C – идентификатор робота, за которого голосует V, W – вес лидера, за которого голосует V, W_{my} – собственный вес, P – расстояние от лидера до данного робота. Группировка является роем (по классификации групп роботов, представленной в [Карпов, 2016] и в [Kernbach, 2013]). За некоторое конечное время необходимо, чтобы роботы самостоятельно организовали стаю, с лидером во главе для дальнейшего решения задачи информационного обмена, который будет достаточно близко располагаться к топологическому центру роя. Формально, близость к центру, гарантированность процесса выбора и единственность лидера определяются таким же образом, как и в [Воробьев, 2017], т.е. необходимо гарантированно перевести рой из начального состояния (1):

$$S_{swarm}=\{V_1,V_2,\ldots,V_N\},\ \forall V_i\ C_i=0,\ i=1,2,\ldots,N \quad (1)$$

в состояние (2):

$$S_{flock}=\{V_1,V_2,\ldots,V_N\},\ \exists!V_i\ C_i=\alpha_i,\ i=1,2,\ldots,N \text{ и} \quad (2)$$

$$\forall V_j\ C_j=\alpha_i,\ j=1,2,\ldots,N;\ j\neq i;$$

Предположим, что $M=\{0,\alpha_1,\alpha_2,\ldots,\alpha_N\}$, тогда гарантированность процесса выбора можно описать следующим образом (3):

$$\forall N\ \exists S=\{S_{swarm},S_1,\ldots,S_P,S_{flock}\},\ S_i=\{V_1,V_2,\ldots,V_N\}, \quad (3)$$

$$\text{где } \forall V_j\ C_j=M_k,$$

$$i=1,2,\ldots,P;\ j=1,2,\ldots,N;\ k=1,2,\ldots,N+1$$

Если интерпретировать рой как граф G, где роботы являются вершинами, а связь с соседями - ребрами, то близость лидера к центру роя d определяется следующим образом (4):

$$d=\frac{s}{r(G)} \quad (4)$$

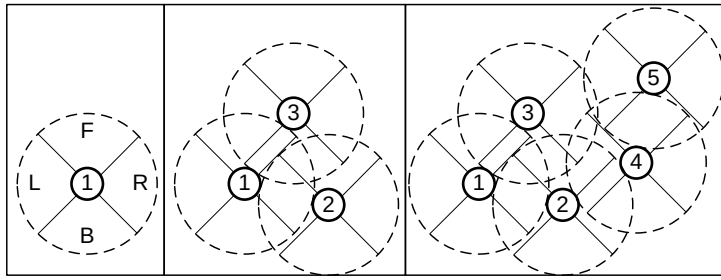
где r – радиус графа G, s – расстояние от центра графа до робота-лидера.

В общем и целом, постановка задачи схожа с постановкой, представленной в [Воробьев, 2017]. Отличием является то, что в общем случае роботы в начальный момент времени могут находиться далеко друг от друга и быть никак не связаны по каналам связи, т.е. $\forall V\ |L| = 0$.

2 Алгоритм решения

Начало процедуры выбора лидера определяется каждым роботом в группе самостоятельно. В этот момент он переключает локальную связь в пассивный режим, т.е. не передает в эфир никакие сообщения, и направляется в (ЦА) – специальное место, которое геометрически является окружностью, радиусом R/2, где R – радиус действия локальной связи робота, координаты которого заранее известны всем роботам. Необходимость в нем возникает из-за того, что роботы заканчивают решение предыдущей задачи несинхронно, следовательно, может возникнуть ситуация, когда одновременно формируется несколько отдельных групп роботов, которые никогда не соединятся, так как при включении робота в группу он прекращает свое движение. Наличие единого “места голосования” позволяет нивелировать эту проблему.

Когда первый робот попадает в ЦА, он переводит локальную связь в активный режим и начинает процедуру выбора лидера. Роботы, которые затем будут приезжать в ЦА, сначала будут активироваться сигналами локальной связи первого робота. Перейдя в активный режим, они смогут активировать прибывающих пассивных членов группы. Это расширяет “зону активации” – объединение областей действия локальной связи активных роботов, попадание внутрь которой, активирует пассивного робота (см. Рис. 1).



а) б) в)

Рис. 1. а) - робот, приехавший в центр ЦА первым. Показаны сектора действия локальной связи б) - два робота, которые были активированы роботом №1 в) - два робота, которые были активированы вторично 2 и 4 роботами. Пуктир - зона действия локальной связи роботов.

Необходимость такого механизма обусловлена тем, что робот может обмениваться данными только с ограниченным числом своих ближайших соседей, т.е. исключается возможность связи “все-со-всеми” при числе роботов большим, чем число каналов локальной связи. Например, если это число равно 4, то он сможет обмениваться информацией только с 4-я

ближайшими соседями, по одному с каждой из сторон (см. Рис. а). Однако, используя соседей как ретрансляторы, робот может опосредованно взаимодействовать с остальными роботами.

Алгоритм выбора лидера представляет собой процесс обмена между роботами своими весами и изменения этих весов в зависимости от того, насколько далеко данный конкретный робот находится от самого удаленного от него робота. При этом играет роль не физическое расстояние между ними, а топологическое – число роботов между ними, так как принимается, что качество локальной связи и скорость передачи по ней данных не зависит то расстояния между роботами, в границах работоспособности локальной связи.

Фактически же, если рассматривать группу роботов как граф, куда произвольно добавляются вершины, а веса ребер одинаковы, то можно сказать, что эта процедура является процедурой динамического определения эксцентриситета всех его вершин. При этом лидером становится вершина с наименьшим эксцентриситетом, т.е. центр графа. Если встречаются вершины с одинаковым эксцентриситетом, то лидером становится вершина с наибольшим или наименьшим α.

При этом распределение весов роботов будет таково, что роботы с меньшими весами будут находится ближе к топологическому центру роя, чем роботы с большими весами (см. Алгоритм 1).

Алгоритм 1. Динамическое определение лидера. При равенстве весов лидером становится робот с наибольшим α.

α – идентификатор робота V, α_L – идентификатор робота-кандидата, L – список соседей робота V, len_{max} – максимальная длина списка L, W – вес робота, за которого голосует V, W_{my} – собственный вес робота V, W_L – вес кандидата, W_{myL} – собственный вес соседа, P – расстояние до робота-кандидата, за которого голосует V, P_L – расстояние от робота-соседа, до кандидата в лидеры, C – идентификатор робота, за которого голосует V, buffer – буфер входных сообщений робота V от соседей в списке L, S – “пройденный путь” сообщения, т.е. число роботов, его ретранслировавших, включая отправителя. С помощью него каждая вершина корректирует свой вес/эксцентриситет.

Начало
$W_{my} = 0$
если робот V не в стае **то**
    цикл пока не конец buffer **то**
        если сообщение от маяка ЦА **то**
            C= α; W=W_{my};P = 0 //V - лидер
        иначе
            $W_{my} = W_{my}+1$

если (($W_{my} < W_L$) **или** (($W_{my} == W_L$) **и** ($\alpha > \alpha_L$)))
то
C= α;W= W_{my};P = 0//V - лидер
иначе
C= α_L;W= W_L;P = P_L+1 //V - подчиняется
Отослать соседям C,W,W_{my},P, S=0
иначе
цикл пока не конец buffer **то**
если buffer - сообщение от лидера **то**
если (($W_{my} < W_L$) **или** (($W_{my} == W_L$) **и** ($\alpha > \alpha_L$)))
то
C= α;W= W_{my};P = 0//V - лидер
иначе
C= α_L;W= W_L;P = P_L+1 //V - подчиняется
Отослать соседям C,W,W_{my},P
иначе
S=S+1
если $W_{my} > S$ **то**
$W_{my} = W_{myL}$+1
Отослать соседям α_L,W_{my},S, P
Конец

3 Эксперименты

Как вычислительные так и реальные эксперименты использовали единый программный базис для получения экспериментальных данных. Его основой является система управления отдельным роботом, а также сопутствующие узлы, осуществляющие коммутацию портов локальной связи, сбор статистики и т.д., при создании которых использовался механизм узлов и топиков операционной системы ROS [ROS Kinetic, 2018]. Отличием является то, что в случае реальных экспериментов использовались навигационные данные (положение, угол поворота) реальных роботов на полигоне. В случае вычислительных экспериментов эти данные брались из конфигурационного файла.

Также для вычислительных экспериментов не моделировался процесс движения роботов. Это было сделано для того, чтобы нивелировать связанный с этим процессом фактор, который существенно влияет на итоговые показатели времени выполнения алгоритма. Иначе говоря, время, за которое будет выбран лидер для определенного числа роботов, безусловно зависит от того, насколько быстро они все достигнут ЦА.

Однако сам процесс выбора лидера не зависит от скорости прибытия роботов в ЦА. В связи с этим был предложен вариант вычислительных экспериментов, где роботы уже сформировали структуру, похожую на статический рой, однако все они находятся в пассивном режиме, т.е. ∀V |L| = 0. Затем произвольным образом инициируется один из этих роботов, как если бы он сам достиг ЦА, а он уже активирует своих соседей, те – своих и т.д., что имитирует их приближение к ЦА. При этом остальные роботы расположены вне ЦА, но в радиусе действия локальной связи инициированного робота.

3.1 Вычислительные эксперименты

При проведении вычислительных экспериментов использовалась возможность промоделировать работу алгоритма для существенного большего числа роботов N=25,50,…,150 с шагом в 25, чем в реальности. Для каждого значения N проводилось 100 вычислительных экспериментов с разной расстановкой роботов. Измерялось два параметра: T - время выполнения алгоритма, d - относительная близость лидера к центру графа этой группы.

На рис. 2 видна линейная зависимость O(N) среднего и максимального времени выполнения алгоритма в зависимости от числа роботов N.

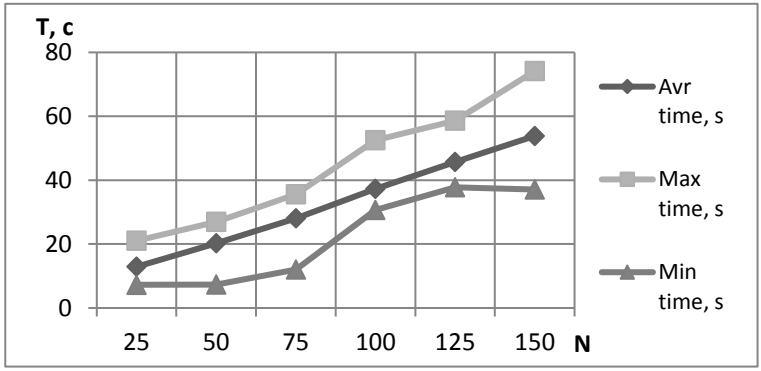


Рис. 2. Зависимость времени выполнения T алгоритма от числа роботов N

Вместе с тем, если сравнивать с алгоритмами, которые равновероятно выбирают лидера в группе, например, с [Karpov et al., 2015], то при одинаковой временной сложности в абсолютных значениях время выполения предлагаемого алгоритма выше.

На Рис. 3, то можно увидеть, что среднее отношение расстояния от центра графа до робота-лидера к радиусу графа Avr dev находится в пределах от 0.1 до 0.12, что говорит о том, что в среднем лидер отстоит от

центра графа не более чем на 10-12% радиуса графа, описывающего конфигурацию группы. Максимальное отклонение доходят до значения 0.63 для N=75, однако это связано непосредственно с формой группы. Дело в том, что при вычислении радиуса графа учитывается непосредственное расстояние между вершинами, а не топологическое. В связи с этим могут быть ситуации, когда вершина является радиусом графа, находясь с краю группы. Алгоритм же учитывает форму группы, что может давать существенные максимальные отклонения.

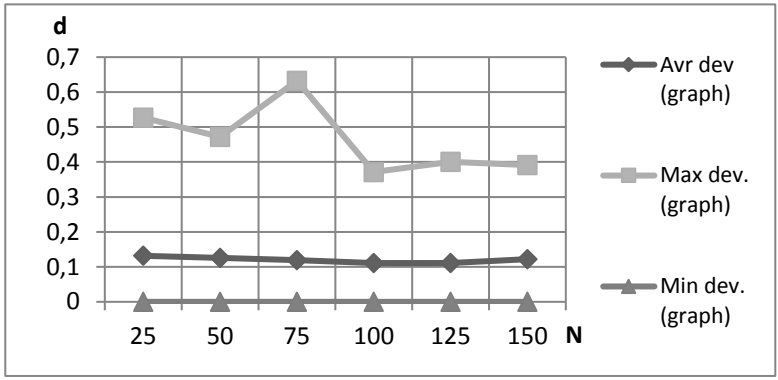


Рис. 3. Зависимость параметра d от числа роботов N

Если сравнивать полученные данные с данными, получаемыми от алгоритмов с равновероятностным выбором лидера, то можно отметить, что первые существенно чаще выбирают лидером робота, который находится в центре группы или не более чем на расстоянии 1-2 роботов от него. Например, для N=25 – 95%, а для N=150 – 85%.

3.2 Эксперименты на реальной группировке роботов

Работа алгоритма была опробована на реальной группировке роботов YARP-2 на лабораторном полигоне. Использовалось 6 роботов в 20 экспериментах с различными расстановками. Как и было отмечено, основной вклад во время выбора лидера дает время, необходимое роботам, чтобы собраться в ЦА. В данном случае общее время работы $T \approx 10\text{-}11$ с. Оценка d для такого числа не показательна, однако в расстановках роботов, которые дают в дальнейшем четко выраженного робота в центре, именно он и становился лидером группы.

Заключение

В работе предложен алгоритм динамического выбора лидера в группе роботов с локальным взаимодействием, который позволяет учитывать топологию получаемой группы. В основе алгоритма лежит механизм обмена весами между роботами группы. Рассматривая схему соединений роботов по их каналам локальной связи в виде графа, отдельного робота можно интерпретировать как вершину графа, а его вес как эксцентриситет этой вершины. Таким образом, с помощью предлагаемого алгоритма лидером становится робот с минимальным эксцентриситетом, который считается динамически и локально для каждой вершины.

Эксперименты и на модели, и на реальной группировке роботов показали работоспособность предлагаемого механизма, приемлемую временную сложность порядка O(N), где N – число роботов.

Список литературы

[Gan Chaudhuri et al., 2015] Gan Chaudhuri S., Mukhopadhyaya K. Leader election and gathering for asynchronous fat robots without common chirality // J. Discret. Algorithms. 2015. (33). P. 171–192.

[Canepa et al., 2007] Canepa D., Potop-Butucaru M. Stabilizing flocking via leader election in robot networks // Proceedings of the 9h international conference on Stabilization, safety, and security of distributed systems. 2007. P. 52–66.

[Dieudonné et al., 2010] Dieudonné Y., Petit F., Villain V. Leader election problem versus pattern formation problem // Distribute computing. 2010. P. 267–281.

[Flocchini et al., 2008] Flocchini P., Prencipe G., Santoro N., Widmayer P. Arbitrary pattern formation by asynchronous, anonymous, oblivious robots // Theor. Comput. Sci. 2008. № 1–3(407). P. 412–447.

[Karpov et al., 2015] Karpov V., Karpova I. Leader election algorithms for static swarms // Biol. Inspired Cogn. Archit. 2015. (12). P. 54–64.

[Kernbach, 2013] Kernbach S // Handbook of Collective Robotics: Fundamentals and Challenges. 2013. 962 p.

[Lynch et al., 1993] Lynch N.A., Patt-Shamir B. Distributed Algorithms // 1993. 440 p.

[Santoro, 2007] Santoro N. Design and analysis of distributed algorithms. //2007. 589 p.

[Воробьев, 2017] Воробьев В.В. Алгоритмы выбора лидера и кластеризации в статическом рое роботов // Мехатроника, автоматизация, управление. 2017. № 3(18). С. 166–172.

[Карпов, 2013] Карпов В.Э. Управление в статических роях. Постановка задачи // VII-я Международная научно-практическая конференция «Интегрированные модели и мягкие вычисления в искусственном интеллекте». // 2013. С. 730–739.

[Карпов, 2016] Карпов В.Э. Модели социального поведения в групповой робототехнике // Управление большими системами. 2016. № 59. С. 165–232.

[ROS Kinetic, 2018] ROS Kinetic documentation. - http://wiki.ros.org/kinetic

УДК 004.89

РЕШЕНИЕ ЗАДАЧИ ФОРМИРОВАНИЯ СТРОЯ БПЛА С ПРИМЕНЕНИЕМ НЕЙРОННОЙ СЕТИ И СИСТЕМЫ ПРАВИЛ

М.В. Хачумов (*khmike@inbox.ru*)
Федеральный исследовательский центр «Информатика и управление» РАН, Москва

Аннотация. Дана постановка задачи безопасного формирования заданного строя для группы беспилотных летательных аппаратов (БПЛА) типа квадрокоптеров, в возмущенной воздушной среде. Для построения опорного плана решения предлагается применить вычислительную схему на основе искусственной нейронной сети Кохонена. Безопасность группового полета обеспечивается путем применения специальных правил управления. Выполнены экспериментальные исследования по моделированию решения задачи формирования строя с учетом ветровых возмущений в среде MATLAB Simulink.[1]

Ключевые слова: БПЛА, строй, нейронная сеть Кохонена, правила управления.

Введение

Строем (образованием или формацией) будем называть требуемое расположение беспилотных летательных аппаратов (БПЛА) на плоскости или в пространстве. В зависимости от того, каким образом задан целевой строй, строевая задача может иметь различную постановку. В простейшем случае целевой строй может быть задан в виде координат целевых положений БПЛА. Методы построения плоских и пространственных формаций в группах квадрокоптеров рассмотрены в ряде современных отечественных и зарубежных работ [Дьяченко, 2012], [Иванов, 2016], [Guzey, 2016], [Xue et al., 2016]. В случае БПЛА самолетного типа, позиции строя должны быть заняты всеми летательными аппаратами одновременно с определенной скоростью. Решение такой задачи рассмотрено автором в работе [Khachumov et al., 2018] и разбивается на два этапа. На первом оно сводится к решению задачи о назначениях, т.е. распределению БПЛА

[1] Работа выполнена при частичной финансовой поддержке РФФИ (проекты № 17-29-07003, 18-07-00025).

группы по целевым положениям и связано с выбором критерия или матрицы стоимости (штрафов). На втором этапе решается задача безопасного движения БПЛА к своим назначенным целевым положениям. В этом случае полезной информацией может служит матрица расстояний между центрами летательных аппаратов [Иванов, 2016], рассчитываемая через определенные интервалы времени, на основе которой осуществляется прогнозирование времени и места возможного столкновения БПЛА при простейших прямолинейных движениях. При обнаружении мест возможного столкновения решается задача устранения конфликтной ситуации. Различные аспекты разрешения коллизий путем перепланирования и введения задержек подробно рассмотрены в работе [Андрейчук и др., 2016].

В настоящей работе расчет траекторий движения БПЛА типа квадрокоптеров к своим целевым позициям осуществляется с применением искусственной нейронной сети (ИНС) Кохонена [Хачумов, 2013]. В процессе движения БПЛА по рассчитанным траекториям под воздействием ветровых нагрузок возможно возникновение коллизий, средством разрешения которых служат специальные правила управления, направленные на изменение скоростей и направлений сближения.

1 Постановка задачи формирования строя

Рассмотрим следующую постановку задачи формирования строя. Пусть n идентичных БПЛА $B=\{b_1,\ldots,b_n\}$ располагаются произвольным образом в некоторой ограниченной области. Состояние БПЛА b_i в момент времени t определяется его координатами $(x_i(t), y_i(t), z_i(t))$, скоростью движения $v_i(t)$ и углами тангажа $\theta_i(t)$ и рыскания $\psi_i(t)$. Предполагается, что известны целевые положения $s_1(x_1^s, y_1^s, z_1^s),\ldots, s_n(x_n^s, y_n^s, z_n^s)$ аппаратов в строе S. Необходимо перевести динамическую систему из заданного состояния $x_i(0), y_i(0), z_i(0), v_i(0), \theta_i(0), \psi_i(0)$, определяющего исходное расположение БПЛА, в целевое состояние $x_i(T), y_i(T), z_i(T), v_i(T), \theta_i(T), \psi_i(T)$ за время T, при этом $\forall i, j = \overline{1,n}$, $(x_i(T), y_i(T), z_i(T)) \in S$, $(x_i(T), y_i(T), z_i(T)) \neq (x_j(T), y_j(T), z_j(T))$. Причем движение группы БПЛА к целевым позициям должно быть безопасным. Упрощенная модель движения БПЛА при отсутствии возмущений принимает вид $\dot{x}_i = v_i \cos\theta_i \cos\psi_i$; $\dot{y}_i = v_i \sin\theta_i$; $\dot{z}_i = v_i \cos\theta_i \sin\psi_i$.

Введем геометрическую модель БПЛА как сферу радиуса R (с некоторым запасом, учитывающим геометрию аппарата). Тогда безопасное расстояние $d_{ij}(t)$ между двумя летательными аппаратами будет

определяться величиной $d_{ij}(t) \geq 2R$. Для решения задачи образования заданного строя из случайно расположенного множества летательных аппаратов предлагается применить вычислительную схему на основе модифицированной ИНС Кохонена с набором метрик [Хачумов, 2013]. В результате работы нейронной сети для каждого БПЛА в группе строится свой безопасный маршрут движения, который он должен отработать с применением группы специальных правил.

2 Формирование опорного плана решения

Предложенная нейронная сеть [Хачумов, 2013] служит для решения задачи рационального перемещения БПЛА из заданного случайного размещения в целевое. Структура такой сети показана на Рис. 1.

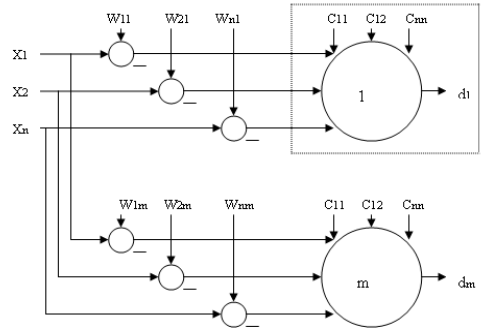


Рис. 1. Нейронная сеть Кохонена с универсальной метрикой

Структура универсальной сети Кохонена характеризуется наличием у каждого нейрона дополнительной памяти для хранения коэффициентов матрицы ковариаций c_{ij}, что необходимо при использовании альтернативных метрик измерения расстояния: Махаланобиса и Евклида-Махаланобиса [Хачумов, 2013].

Первый слой сети имеет три входа, что соответствует числу координат. Второй слой имеет n нейронов, что соответствует числу БПЛА. Начальные весовые коэффициенты сети устанавливаются следующим образом $(w_{1j}, w_{2j}, w_{3j}) = (x_j(0), y_j(0), z_j(0))$. Весовые коэффициенты настройки $(w_{1j}(t), w_{2j}(t), w_{3j}(t))$ определяют топологию сети в момент времени t и интерпретируются в сети Кохонена как значения координат БПЛА $(x_j(t), y_j(t), z_j(t))$. Затем осуществляется подача на вход сети в случайном порядке координат точек формации S. Эти координаты интерпретируются как входной вектор (x^s, y^s, z^s). Для дальнейшего решения задачи

используется типовой алгоритм работы сети Кохонена, но с включением проверки на каждом шаге величины сближения БПЛА.

Алгоритм настройки сетевой модели

1. На вход сетевой модели подается очередной вектор. Для каждого нейрона j вычисляется расстояние Евклида между его вектором весов и входным вектором
2. Определяется нейрон-победитель с минимальным расстоянием d_j;
3. Для нейрона-победителя производится модификация его весовых коэффициентов по правилу $w_{ij}(t+1) = w_{ij}(t) + \eta(t)(e_i(t) - w_{ij}(t))$, где: $w_{ij}(\mathrm{t})$ – значение весового коэффициента, связывающего вход i сети с нейроном j, $\eta(\mathrm{t})$ – норма обучения, $0 < \eta(\mathrm{t}) \le 1$, $e_i(\mathrm{t})$ – значение входа в момент времени t.
4. Осуществляется проверка безопасности. Если условие $d_{ij}(t) \ge 2R, \ \forall i, j$ выполнено, то происходит обновление весовых коэффициентов нейрона-победителя, что определяет его перемещение; иначе на данной итерации нейрон-победитель не рассматривается, перейти к п.2.
5. Перейти к п.1

При условии существования опорного плана его получение гарантируется. На каждой итерации происходит «подтягивание» выбранных БПЛА к целевым позициям, при этом шаг работы ИНС привязан к модельному (или реальному) времени. Все моменты времени, связанные с перемещениями БПЛА являются известными, что позволяет на каждом шаге определять местоположение всех объектов и расстояния между ними с учетом введенной геометрической модели. После того как траектории движения всех БПЛА получены, необходимо обеспечить их отработку с учетов ветровых возмущений и возможных коллизий.

3 Отработка траекторного движения в сложных условиях

Пусть эталонная траектория каждого летательного аппарата $b_i \in B$, $i = 1,...,N$ задана движением объекта $c_i \in C$, $C = \{c_1,...,c_N\}$, называемого далее «псевдоцель», и представлена последовательностью из M_i опорных точек $(x_{ij}, y_{ij}, z_{ij}), j = 1,...,M_i$. Псевдоцель c_i осуществляет движение между соседними опорными точками траектории с эталонной скоростью $v_i^{(c)}$ и углами $\theta_{ij}^{(c)}$, $\psi_{ij}^{(c)}$, а БПЛА преследует ее, руководствуясь выбранной стратегией. На основе предыдущего этапа решения задачи известно желаемое время прохождения опорных точек

$t_{ij}^{(c)}$, $i=1,...,N$, $j=1,...,M$ и всего маршрута в целом $T_i^{(c)}$. В результате ветровой нагрузки возможно отклонение каждого летательного аппарата от своего маршрута, причем существенное, что может вызывать столкновение. Скорость v_i, углы тангажа θ_i и рыскания ψ_i БПЛА b_i устанавливаются системой управления в допустимых пределах.

Рассмотрим постановку задачи следования группы БПЛА по заданному маршруту. Пусть $B_i(t)$ и $C_i(t)$ − координаты БПЛА b_i и соответствующей цели c_i, а $d(B_i(t),C_i(t))$ − расстояние между ними в момент времени t. Задача заключается в построении такого управления для каждого БПЛА в группе $U_i(t)=(v_i(t),\theta_i(t),\psi_i(t))$ на временном отрезке $[0,T_i]$, что $\int_{t=0}^{T_i} d(B_i(t),C_i(t))dt \to \min$ при ограничении: $d(B_i(t),B_j(t)) \geq 2R$, $\forall i,j,\ i \neq j$. Предлагается решение, основанное на действиях, имитирующих поведение пилота и заключающееся в выборе стратегий, реализуемых наборами правил в условиях установленных ограничений на управление и действующих возмущений [Abramov et al., 2015]. Рассмотрим некоторые стратегии управления БПЛА в группе.

Стратегия 1 (движение по точкам) для каждого БПЛА b_i заключается в коррекции движения по текущему отклонению от заданной траектории следования псевдоцели. При этом должны быть пройдены все точки с минимальным отклонением по времени $\Delta_i^{(T)} = \left|T_i - T_i^{(c)}\right| \to \min$.

Стратегия 2 (сближение с псевдоцелью) осуществляет «параллельное сближение» [Abramov et al., 2015] БПЛА с псевдоцелью и предполагает вычисление углов тангажа и рыскания для прогнозирования их места встречи. Точное прохождение БПЛА через опорные точки не требуется.

Приведем применяемые продукционные правила в порядке приоритета их исполнения:

Правило 1 (устранение опасного сближения): если расстояние $d(B_i(t),B_j(t)) < 2R$, то для b_i и b_j применить Стратегию 2.

Правило 2 (режим следования): если отклонение и время отработки отклонения для b_i и соответствующей псевдоцели c_i не превышает наперед заданных пороговых значений, то применить Стратегию 1.

Правило 3 (режим упреждения): если отклонение или время отработки отклонения для b_i и соответствующей цели c_i превышают пороговые значения, то применить Стратегию 2.

Ветровые нагрузки могут существенно влиять на траектории движения БПЛА, вызывая ситуации опасного сближения. Дополнительным путем их разрешения, корме указанных выше правил, является применение

эвристических правил расхождения, аналогичных принятым в авиации и судоходстве. Правила более детально изложены в работе автора [Khachumov, 2018].

4 Экспериментальные исследования

Выполнено моделирование решения задачи формирования строя в группе БПЛА с учетом математических моделей летательных аппаратов, принятой стратегии решения задачи безопасного движения и ветровой нагрузки [Abramov et al., 2015]. На Рис 2. показаны траектории движения пяти БПЛА формирующих заданный строй в возмущенной воздушной среде.

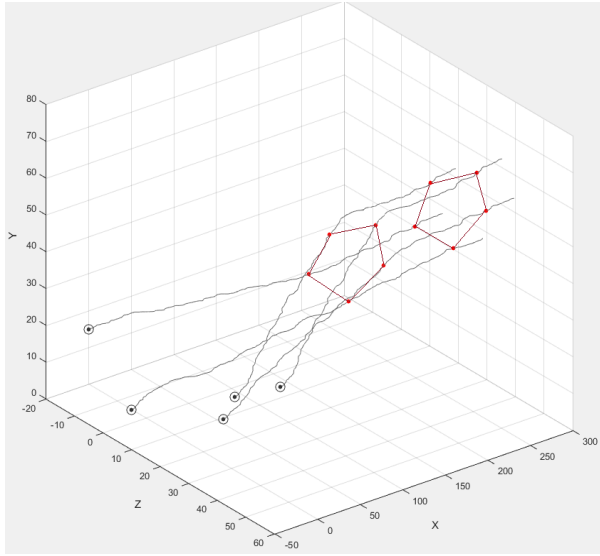


Рис.2. Формирование группой из пяти БПЛА заданного строя

Из рисунка видно, как БПЛА перемещаются в заданные позиции и формируют необходимый строй. Разработанная имитационная модель в системе MATLAB Simulink содержит специальный модуль интеллектуального управления, реализующий стратегии и правила управления для оперативного реагирования на изменения внешней среды.

Заключение

Рассмотрен подход к решению задачи безопасного формирования строя для группы беспилотных летательных аппаратов. На первом этапе

формируются эталонные траектории движения БПЛА, что подразумевает определение опорного плана перемещения для достижения назначенных позиций с применением ИНС Кохонена. На основе сетевой модели можно решать задачи формирования строя БПЛА и его реконфигурации, решать другие актуальные прикладные задачи. Для безопасного движения группы БПЛА по установленным траекториям предлагается подход, основанный на преследовании «псевдоцелей», с применением правил, направленных на снижение риска столкновений. Ожидается, что бортовые алгоритмы управления на основе предложенных стратегий и правил обеспечат приемлемую точность решения задачи безопасного формирования строя для небольших БПЛА при ветровых нагрузках.

Список литературы

[Андрейчук и др., 2016] Андрейчук А.А., Яковлев К.С. Метод разрешения конфликтов при планировании пространственных траекторий для группы беспилотных летательных аппаратов // Третий Всероссийский научно-практический семинар «Беспилотные транспортные средства с элементами искусственного интеллекта» (БТС-ИИ-2016, 22-23 сентября 2016 г., г. Иннополис, Республика Татарстан, Россия): Труды семинара. – М: Изд-во «Перо». 2016.

[Дьяченко, 2012] Дьяченко А. А. Задача формирования строя в группе БПЛА // Известия Южного федерального университета. Технические науки. 2012. № 3(128).

[Иванов, 2016] Иванов Д. Я. Методы построения пространственных формаций в группах беспилотных летательных аппаратов типа квадрокоптеров. – Дисс. канд. техн. наук, Таганрог: Южный федеральный университет. 2016.

[Хачумов, 2013] Хачумов М.В. Сетевая модель кластерного анализа // Прикладная физика и математика. 2013. № 10.

[Abramov et al., 2015] Abramov N. S., Makarov D. A., Khachumov M. V. Controlling flight vehicle spatial motion along a given route // Automation and Remote Control. 2015. № 6(76).

[Guzey, 2016] Guzey H.M. Adaptive consensus based formation control of unmanned vehicles. – Doctoral dissertation, Missouri University of Science and Technology, 2016.

[Khachumov, 2018] Khachumov M. V, Khachumov V.M. On the Safe Achieving the Required Formation Shape by Multiple UAVs (Planar Case) // Материалы Всероссийской конференции с международным участием “Информационно-телекоммуникационные технологии и математическое моделирование высокотехнологичных систем” (16–20 апреля 2018, Москва, РУДН). 2018.

[Khachumov et al., 2018] Khachumov M.V., Khachumov V.M. The model of UAV formation based on the uniform allocation of points on the sphere // MATEC Web Conf. 13th International Scientific-Technical Conference on Electromechanics and Robotics “Zavalishin’s Readings”, St. Petersburg, Russia. 2018.

[Xue et al., 2016] Xue R. and Cai G. Formation flight control of multi-UAV system with communication constraints // Journal of aerospace technology and management. 2016. № 2(8).

УДК 004.896

АНАЛИЗ СВЯЗНОСТИ КАРТЫ СТАЕЙ РОБОТОВ С КОММУНИКАЦИЕЙ

В.Е. Павловский (vlpavl@mail.ru)
Институт прикладной математики им. М.В. Келдыша РАН, Москва

В.В. Павловский (vlpavl2000@mail.ru)
Российский Экономический Университет им. Г.В. Плеханова, Москва

М.В. Андреева (point15@ro.ru)
Московский автомобильно-дорожный институт (технический университет) МАДИ

Аннотация Описывается алгоритм, решающий задачу анализа связности карты, которая строится распределенной информационной системой роботов. Такая задача может быть реализована стаей летающих роботов-разведчиков, например, для контроля возможности прохода между препятствиями группой наземных роботов, которым сообщается разведанная информация. Анализ связности карты выполняется на основе специальной перенумерации областей связности, которая реализуется в обмене данными между роботами-разведчиками. 1 Работа является переработанной версией статьи, представленной в журнале Мехатроника. Автоматизация. Управление.

Ключевые слова: мобильный робот, картирование, карта, связность карты

Введение. Постановка задачи

Задача картирования местности, т.е. построения карты, является одной из центральных при обеспечении движения мобильных роботов (МР) в сложной или неизвестной среде. Подобные задачи составляют широкий класс задач информационного обеспечения МР, к таким задачам относятся задачи исследования и составления карты неизвестной местности,

[1] Работа выполнена при поддержке РФФИ, проекты 16-08-00880-а, 16-01-00131-а, 15-07-07483-а, 16-29-04412-офи-м, и при поддержке Программы РАН 1.31, раздел "Актуальные проблемы робототехники"

ориентирования и навигации на местности, задачи контроля либо инспекции обследованных районов и другие аналогичные. В связи с важностью таких задач к настоящему времени появилось большое число систем, в целом обеспечивающих и поддерживающих их решение. Это прежде всего системы, отображающие на карте рельеф местности и препятствия, такие системы созданы для роботов, для групп роботов [Зенкевич и др., 2007], [Шварц и др., 2016], или как помощники человеку, работающему в естественной среде. К последним относится, например, экспериментальная носимая интеллектуальная система на базе сенсора типа Kinect и лазерного дальномера , созданная в MIT, США. При этом значительное число современных исследований выполнено как создание систем одновременного картирования и навигации (локализации) роботов – систем класса SLAM (Simultaneous Localization And Mapping). Однако важно отметить, что практически все такие системы ограничиваются рассмотрением внешних контуров (габаритов) препятствий, тогда как в ряде прикладных задач также важным является исследование внутренней "топологии" препятствий, особенно при протяженных их размерах.

Настоящая работа продолжает направление, начатое авторами в [Павловский и др., 2015]. Как и ранее, рассматривается стая роботов, в которой для всех роботов задана одинаковая и при этом достаточно простая модель поведения. При этом интерес представляет возможность синтеза сложного поведения стаи в целом и решения стаей содержательных задач на основании относительно простых правил для отдельных роботов. В предыдущей работе моделировалось и исследовалось движение стаи к целям и обход препятствий, а задачей данной работы является исследование стаей окружающей среды; в частности - анализ карты местности. Подобные задачи хорошо подходят для распределенных систем.

В настоящей работе для анализа выбрана одна из фундаментальных топологических характеристик карты - связность. Предполагается, что на местности имеются объекты - препятствия или ориентиры, и роботы снабжены соответствующими датчиками для их определения. Задачей стаи является определение количества областей связности, на которые карта разбивается этими объектами.

Эту задачу можно рассматривать как шаг к распределенному исследованию и распознаванию изображений. Например, хорошо известные изображения (геоглифы) в пустыне Наска имеют огромные размеры, и с земли целиком не видны - понимание этих изображений появилось только после аэрофотосъемки. Поэтому в качестве последнего примера в работе модель применяется к фотографии одного из изображений в пустыне Наска.

Анализ связности карты

Решение этой задачи разбивается на два этапа:

1. Распределение стаи в исследуемой области. Стая выдвигается в намеченную область и рассеивается, образуя большое облако.
2. Анализ геометрических свойств карты. На этом этапе стая считается неподвижной. Предполагается, что после выполнения задачи стая возвращается к месту отправления и "сдает" выполненную работу, т.е. передает ее потребителю.

Этап 1. Распределение стаи

На этом этапе роботы должны выдвинуться в исследуемую область и образовать рассеянное облако. Для настройки нужного поведения стаи использовалась развитая в предыдущей работе [Павловский и др., 2015] идея "псевдосил", которые "подталкивают" роботов в нужном направлении. Но это не физические взаимодействия, а поведенческие правила - "желание" роботов двигаться в том или ином направлении. И поэтому, в отличие от физических сил, псевдосилы определяют не ускорения, а скорости роботов. Таким образом упрощается управление и не возникают нежелательные режимы типа колебательных. В то же время, подбором этой функции можно программировать роботов для синтеза нужного поведения стаи в целом.

Для образования рассеянного облака псевдосила, действующая на робот номер *k*, задается соотношением

$$\vec{F} = -\sum_j f\left(\left|\vec{r}_j - \vec{r}_k\right|\right)\frac{\vec{r}_j - \vec{r}_k}{\left|\vec{r}_j - \vec{r}_k\right|} + K_3\frac{\vec{r}_T - \vec{r}_k}{\left|\vec{r}_T - \vec{r}_k\right|} + K_4\frac{\vec{r}_T - \vec{r}_A}{\left|\vec{r}_T - \vec{r}_A\right|} \qquad (1)$$

Первое слагаемое представляет собой сумму по всем остальным роботам и определяет "отталкивание" роботов друг от друга, которое и приводит к нужному рассеиванию стаи. Функция *f(r)* задает величину отталкивания и вычисляется как максимум из двух функций

$$f(d) = \max\left(K_1 f_1(d), K_2 f_2(d)\right)$$

$$f_1(d) = \begin{cases} 1 & d \le d_1 - \Delta \\ \left(d_1 - d\right)/\Delta & d_1 - \Delta < d \le d_1 \\ 0 & d > d_1 \end{cases}$$

$$f_2(d) = \begin{cases} \left(1 - \frac{d}{d_2}\right)^2 & d \le d_2 \\ 0 & d > d_2 \end{cases}$$

причем $d_1 > d_2$, $K_1 < K_2$. Как видно из формул, первая величина $K_1 f_1$ постоянна до определенного радиуса d_1, затем спадает до нуля. Это основная псевдосила, определяющая рассеивание стаи. Вместе с более слабым притяжением к центру, которое задается двумя последними слагаемыми формулы (1), это отталкивание приводит к тому, что соседние роботы распределяются примерно на расстоянии d_1 друг от друга.

Вторая величина $K_2 f_2$ действует в малом радиусе d_2, но быстро квадратично растет. Эта псевдосила, введенная в предыдущей работе [Павловский и др., 2015], препятствует столкновениям роботов, если в процессе движения по каким-либо причинам (например, отталкивание от других роботов стаи) два робота слишком сильно сближаются.

Как видно из этих формул, сила отталкивания, определяемая функцией $f(r)$, равна нулю за пределами некоторого радиуса, поэтому сумма в первом слагаемом формулы (1) фактически ведется только по роботам в определенной окрестности и может определяться каждым роботом локально.

Второе и третье слагаемое в формуле (1) задают "притяжение" стаи к цели – к исследуемой области, центр которой задан координатами r_T (*Target*). Второе слагаемое описывает притяжение к цели непосредственно выбранного робота. Но использование одного этого слагаемого привело к следующей проблеме: при большой константе притяжения K_3 облако роботов получается сильно неравномерным - сгущается к центру, поскольку роботы по краям тянутся к центру и своим отталкиванием дополнительно прижимают к центру роботы в середине. А при малой константе K_3 роботы выдвигаются к цели слишком медленно.

Поэтому в систему было добавлено последнее слагаемое, описывающее притяжение к цели всей стаи. В этой формуле r_A - центр "масс" стаи, вычисляемый как среднее арифметическое координат всех роботов, $r_A = \frac{1}{n}\sum_j r_j$.

Это слагаемое помогает "подтолкнуть" стаю к месту выполнения задачи, не вызывая деформации облака. При этом коэффициенты выбираются так, что K_3 мало и $K_4 > K_3$. Второе слагаемое в формуле (1) с малым K_3 все равно необходимо, иначе отталкивание роботов может привести к фрагментации облака. Слабое притяжение всех роботов к центру обеспечивает цельность облака и примерно круглую форму, не мешая рассеиванию на заданное

расстояние. Тем самым второе слагаемое в (1) удерживает все облако от "рассыпания", преодолевая взаимные отталкивания и сохраняя общую конфигурацию облака.

Завершение первого этапа можно определять либо по тому, что роботы перестали заметно смещаться (на моделировании видно, что стая после выдвижения и рассеивания достаточно быстро стабилизируется), либо просто по таймеру, выделяя на этот этап определенное время, по истечении которого роботы останавливаются и переходят ко второму этапу.

Этап 2. Анализ карты

На этом этапе предполагается, что роботы рассеялись на местности и уже не движутся. После этого каждый робот определяет, видит ли он картографируемые объекты в определенной окрестности. Как сказано выше, предполагается, что для этого роботы оснащены некоторыми датчиками, от которых описываемая модель абстрагируется. Вместо этого для каждого робота задается фиксированная окрестность радиуса r_1 и считается, что роботу известно, пересекается ли эта окрестность с одним из объектов на местности.

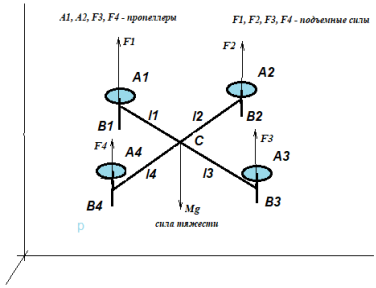
На основании этой информации анализ связности выполняется с помощью следующего алгоритма, основанного на волновом алгоритме (алгоритме маршрутизации - используется волновое распространение данных, номеров роботов):

1. Каждый робот хранит числовую переменную - номер. В начале всем роботам присваиваются различные номера от 1 до N, где N - количество роботов в стае.
2. После рассмотрения препятствий те роботы, которые видят препятствие, заменяют свой номер на –1 и далее в анализе не участвуют.
3. Каждый робот используя функцию коммуникации связывается со всеми роботами в определенной окрестности заданного радиуса r_2. Пусть номер данного робота x, а номер его соседа y. Если $x > 0$ и $y > 0$ и $y < x$, то робот заменяет свой старый номер x на номер соседа y. Иначе говоря, для каждой пары соседних роботов, ни один из которых не видит препятствие, робот с большим номером заменяет свой номер на меньший номер соседа.
4. Операция 3 повторяется, пока номера роботов не перестают меняться.

В процессе применения операций 3 и 4 в каждой области связности постепенно распространяется наименьший из номеров роботов, попавших в эту область - в результате все роботы в области получат этот номер, тогда как в других областях номера всех роботов будут также одинаковыми, но

другими. Таким образом, количество областей связности будет равно количеству различных номеров, оставшихся в стае (не считая номера -1).

В этом алгоритме r_1 и r_2 являются параметрами – это размер области видимости (наблюдения) и размер области соседства роботов, соответственно. Их схема приведена на рис.2. Эта же схема моделировалась в полете коптеров.



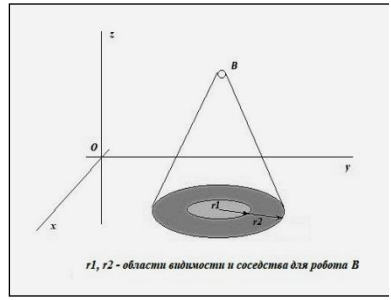


Рис.1. Динамическая модель коптера (слева).
Рис.2. Область видимости и область соседства объектов в стае (справа).

При моделировании оптимальные результаты получились при соотношении радиусов $r_2 = 2r_1$

Для проверки описанного алгоритма было проведено моделирование на ряде примеров - как искусственных модельных препятствий, так и реальных изображений из пустыни Наска. В заключение приведем изображения с результатами работы программы моделирования.

а) Простое кольцевое препятствие

На рис.3 и рис.4 - начальная конфигурация (рис. 3, слева), выдвижение роботов (рис. 3, справа), завершение распространения (рис. 4, слева) и окончание анализа – найдены две области связности (рис. 4, справа):

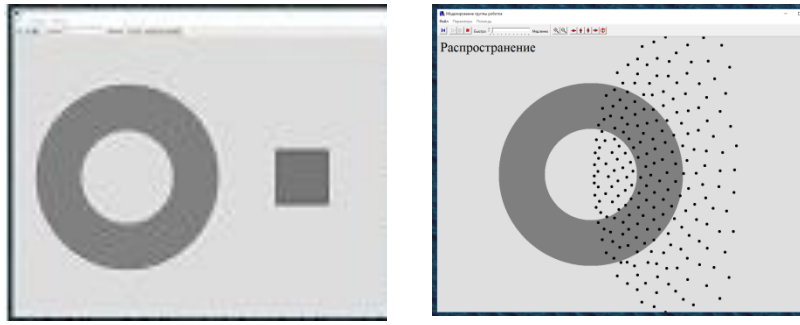


Рис.3. Начальная конфигурация, выдвижение роботов.

Схематично на фазе окончания анализа (рис.4 справа) на роботах изображаются специальные маркеры состояния – они показывают, видит ли данный объект препятствия или определил искомые области связности карты.

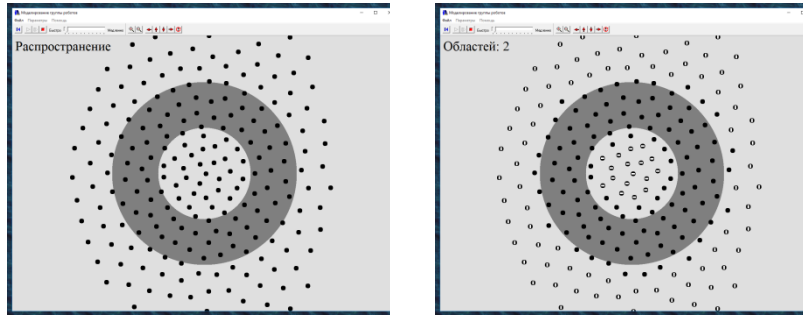


Рис.4. Завершение распространения и окончание анализа.

б) Изображение (геоглиф) из пустыни Наска

На рис. 5 показан результат моделирования обследования изображения из пустыни Наска, выдвижение стаи (слева) и завершение анализа (справа), найдено три области.

На последнем изображении примечателен одиночный робот внизу (показан белой стрелкой), нашедший "лишнюю" область: хотя из этой области есть выход, но он слишком мал, а роботов не так много, их радиусы видимости велики, и поэтому разрешения стаи не хватило, чтобы распознать этот выход.

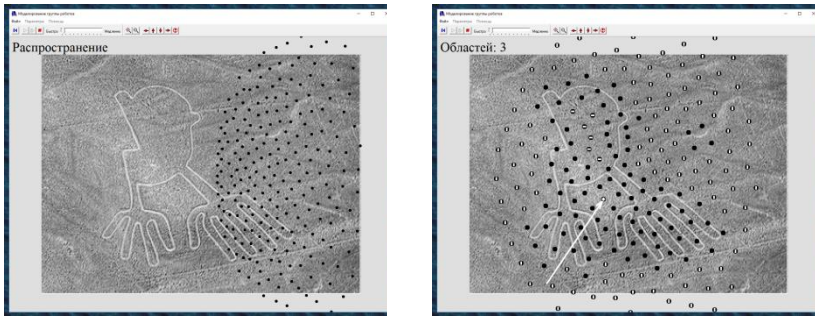


Рис.5. Геоглиф из пустыни Наска.

Заключение

Для экспериментов была соответствующим образом модифицирована программная моделирующая система, описанная в [Павловский и др., 2015]. Все изображения эпизодов моделирования выше в работе, как отмечалось, получены с ее помощью. Проведены серии экспериментов по моделированию созданного алгоритма. Алгоритм показал устойчивое и достаточно эффективное функционирование.

Таким образом, в целом можно отметить, что метод работоспособен, но для распознавания изображений с мелкими деталями требуются стаи (рои) из большего количества роботов с небольшими точными зонами видимости. В развитие описанного исследования предполагается рассмотреть задачу определения такого соответствующего задаче количества информационных роботов.

Список литературы

[Зенкевич и др., 2007] Зенкевич С.Л., Минин А.А. Построение карты мобильным роботом, оснащенным лазерным дальномером, методом рекуррентной фильтрации. // Мехатроника, Автоматизация, Управление. 2007, №8, с.5-12.

[Павловский и др., 2015] Павловский В.Е., Павловский В.В. Математическая модель двумерной гомогенной стаи роботов. // ИИиПР, 2015 г., №4, с. 62-71.

[Шварц и др., 2016] Шварц Д., Куприянов Д.В. Построение карт местности робототехническими системами. // Изв. ВУЗов. ПРИБОРОСТРОЕНИЕ. 2016. Т. 59, № 8, с.695-698.

УДК 004.896

СОЦИАЛЬНЫЕ МОДЕЛИ КОМАНДНОГО ПОВЕДЕНИЯ РЕАКТИВНЫХ РОБОТОВ

А.А. Кулинич (kulinich@ipu.ru)
Институт проблем управления РАН, Москва

Аннотация: Исследуются вопросы формирования и функционирования команд роботов с реактивной архитектурой. Предложены принципы поведения каждого робота команды, основанные на анализе ими состояния среды функционирования роботов, оценки состояния среды с помощью эвристик и не предполагают информационного обмена между роботами.[1]

Ключевые слова: команда роботов, реактивный робот, принципы поведения, социальная модель.

Введение

В теории группового управления интерес представляют роевые подходы к управлению группой роботов. Роевые алгоритмы, основанные на локальном взаимодействии множества однородных роботов (роя роботов), обеспечивают их скоординированное движение, обход препятствий и основывается на принципах, формализующих движение стаи птиц [Reynolds, 1987]. В научной литературе много публикаций о реализации роевых алгоритмов для решения разнообразных задач. Так в работе [Бурдун и др., 2010] приводится 40 алгоритмов, реализующих самое разнообразное движение групп роботов.

Кроме задач движения группы роботов строем, в групповой робототехнике существует ряд типовых задач – задачи патрулирования территории, фуражировки, картирование и др. [Карпов и др., 2019]. Для решения таких задач принципов движения роботов строем Рейнольдса оказывается недостаточным. Поиск принципов поведения группы роботов, которые обеспечивали бы решение конкретной общей задачи, в общем, не тривиальная задача. Обычно в качестве таких принципов выбираются принципы поведения насекомых, животных или людей. Однако, прямой перенос принципов поведения насекомых или людей на поведение искусственных роботов связан с ограничениями технологического

[1] Работа выполнена при частичной поддержке грантов: проект РФФИ ОФИ_м № 16-29-04412.

характера. В работе [Кулинич, 2016] были рассмотрены ряд алгоритмов стайного и командного поведения роботов, которые предполагали обмен информации между роботами о собственных целях и ресурсах. В этой работе сохраняется преемственность, результатов полученных в работе [Кулинич, 2016], однако акцент сделан на разработку принципов поведения так называемых стихийных команд роботов, не предполагающих обмен информацией. Поведение роботов сформулировано в форме принципов, основанных на анализе состояния среды функционирования и собственных целей и ресурсов роботов.

1 Среда функционирования формальных роботов

Рассматривается множество роботов $A=\{R_i\}$, обладающих свойствами (параметрами) $F=\{f_i\}$. Для каждого свойства каждого робота определено упорядоченное множество их возможных значений, $Z=\{Z_i\}$, где $Z_i=\{z_{i1},\ldots z_{iq}\}$, $z_{iq+1} \succ z_{iq}$, $q=0\ldots n\text{-}1$. Кроме этого, определено множество объектов $B=\{b_j\}$, каждый из которых имеет свойства из множества Z свойств роботов. Среда функционирования роботов определяется как прямое произведение множеств значений всех свойств роботов, $SF=\underset{i}{\times} Z_i$.

Вектор значений всех свойств роботов и свойств всех объектов $Y(t) = (Y_1(t),\ldots, Y_n(t),\ldots, Y_{b1}(t),\ldots, Y_{bm}(t))$ определяет состояние среды функционирования, где $Y_1(t)$, $Y_n(t)$ – свойства *i-о* робота; $Y_{b1}(t), Y_{bm}(t)$ – свойства объектов; $Y_i(t)= (z_{1j},\ldots, z_{nb})$, $z_{ij} \in Z_i$, $\forall i$.

Динамика изменения состояния среды функционирования происходит в случаях изменения роботами своих свойств или свойств объектов и представляется как отображение:

$$W\colon Y(t) \rightarrow Y(t+1), \qquad (2)$$

где W – система правил поведения роботов, заданных на множестве возможных состояний среды $W\colon \underset{i}{\times} Z_i \rightarrow \underset{i}{\times} Z_i$; $Y(t)$, $Y(t+1)$ – состояния среды в моменты времени t.

Каждый робот характеризуется следующим кортежем:

$$\langle g_q, r_q, \mu_q(Y_q, g_q), O(r_q)\rangle, \qquad (3)$$

где

1) $g_q = (z_{1j}^{g}, \ldots, z_{nb}^{g})$ – вектор целевых значений робота q, где $g_q \in SF$;

2) $r_q = (z_{1j}^{r}, \ldots, z_{nb}^{r})$ – стратегия достижения цели робота q, где $r_q \in U_q$, $U_q = \underset{i}{\times} Z_i^r$, $Z_i^r \subseteq Z_i$ – ресурсы робота q.

Считается, что робот q применяет стратегию r_q для достижения своей цели g_q, предполагая, что другие роботы никаких действий не совершали.

3) $\mu_q(Y_q(n), g_q)$ – возможность достижения роботом q целевого состояния g_q за счет собственных ресурсов. Пусть в пространстве состояний $(\underset{i}{\times} Z_i)$ определена метрика $\rho(a, b)$, $a,b \in \underset{i}{\times} Z_i$. Тогда возможность достижения цели роботом определяется как близость прогнозной $Y_q(n)$ и его целевой ситуации g_q:

$$\mu_q(Y_q(n), g_q) = \rho(Y_q(n), g_q)^{-1}.$$

По сути, этот показатель определяет потенциальную «силу» каждого робота команды без поддержки других членов команды.

4) $O(r_q)$ – полезность целевой ситуации для робота q. Под полезностью целевой ситуации для робота здесь понимается параметр, характеризующий поведение робота, который используется в процессах формирования команд роботов.

Задача заключается в переводе состояния среды функционирования из текущего положения $Y(t)$ в целевое состояние $Y^*(n)$ путем изменения собственных свойств роботов $a_i \in A$ и свойств объектов $b_i \in B$. Иначе говоря, задача заключается в изменении свойств объектов или роботов, например, это может быть перемещение объекта или робота на плоскости в некоторую заданную точку. Считается, что зная состояние среды функционирования $Y(t)$, робот, используя ресурсы r_q, пытается достичь своей цели g_q, осознавая при этом возможность достижения цели самостоятельно $\mu_q(Y_q(n), g_q)$ и полезность ее достижения $O(r_q)$.

Отметим, что в литературе рассматриваются два способа организации совместной работы роботов. Это централизованный метод, когда их совместная работа планируется заранее, и каждый робот получает задание на выполнение своей части общей работы, и децентрализованный, когда роботы на основе принципов самоорганизации самостоятельно решают общую задачу. Далее будет рассмотрен децентрализованной метод взаимодействия роботов, основанный на принципах социальной организации в малых социальных группах.

2 Принципы формирования и функционирования стихийных команд реактивных роботов

Минимальное определение интеллектуальных роботов – это перечисление их свойств: автономность – способность действовать самостоятельно; реактивность – способность реагировать на изменение среды; проактивность – способность проявлять активность согласно своим целям; социальность – способность общаться с другими роботами.

Далее мы будем рассматривать реактивных роботов, у которых присутствуют две способности – автономность и реактивность.

Под принципом формирования и функционирования команды роботов мы будем понимать поведение каждого робота, основанное на анализе среды функционирования и имеющейся у робота информации о собственном состоянии.

Основываясь на работах социальных психологов, исследовавших вопросы самоорганизации и сплоченности в малых социальных групп мы будем рассматривать следующие принципы поведения роботов при выполнении совместной работы:

- принцип самостоятельного достижения цели, который означает, что робот пытается самостоятельно достичь цели, независимо от возможности ее достижения;
- принцип взаимной полезности – означает, что робот не способный достичь цели взаимодействует с роботом, который имеет лучшие возможности достижения цели и которому он сам будет полезен;
- принцип «ленивости» - означает, что робот не присоединяется к работе в группе роботов, если эта группа достигает цели этого робота;
- принцип «эгоистичности» - работает в случаях, когда роботы получают «вознаграждение» за достижение цели. Эгоистичные роботы пытаются достичь цели и получить вознаграждение, даже в случаях, если их возможности достижения цели невелики.

3 Стихийные команды реактивных роботов

Под стихийной командой реактивных роботов будем понимать множество роботов имеющих общую цель, возможность наблюдать за состоянием среды функционирования и не имеющие возможности общаться и координировать свои действия - строить общий план.

Итак, есть группа реактивных роботов R_i, каждый из которых определен кортежем (3). Считаем, что роботы не могут общаться, но они знают состояние среды функционирования $Y(t)$, и могут изменить это состояния, используя для этого собственные ресурсы.

3.1 Принцип самостоятельного достижения цели

Рассмотрим формирование и функционирование группы роботов на основе принципа самостоятельного достижения цели.

$$Y^*(t+1)=W^\circ\, Y(t)\oplus(\bigoplus_i r_i)$$

В этом случае роботы не знают о существовании других роботов их целей и возможностей их достижения. Все роботы применяют одновременно собственную стратегию r_i с целью достичь собственного целевого состояния g_i. Поскольку цели у роботов могут быть разными то,

агрегация стратегий $\bigoplus_i r_i$ не гарантирует, что каждый робот достигнет цели.

Однако, здесь возможны частные случаи.

1. В группе роботов существует робот, стратегия которого доминирует агрегированные стратегии всех остальных роботов. Такого робота будем называть диктатором. Если в группе роботов есть робот-диктатор, то он достигает своей цели, а остальные роботы нет.
2. В группе роботов существуют роботы, имеющие близкие цели, $\forall R_i \in K$, $K \subseteq A$, $\rho(g_i, g_q) \le \varepsilon$, $\forall R_i, R_q \in K$, ε – критерий близости целей. В этом случае образуется команда роботов, и в случае если их агрегированная стратегия доминирует стратегии всех остальных роботов, то команда играет роль робота-диктатора.

3.2 Принцип взаимной полезности

Необходимым условием работы команды на этом принципе являются наличие в группе роботов A, роботов с близкими целями. Эти роботы образую команду, и могут объединять собственные ресурсы. Считаем, что есть группа роботов A (3); есть подмножество K роботов с близкими целями $\forall R_i \in K$, $K \subseteq A$, $\rho(g_i, g_q) \le \varepsilon$, $\forall R_i, R_q \in K$.

Вначале сформулируем формально принцип полезности роботов.

Роботов R_i, R_q с близкими целями будем называть взаимно полезными, если объединение их ресурсов увеличивает возможность достижения общей цели g_Σ. В случае объединения ресурсов роботов коалиции прогноз развития ситуации определиться из выражения (4):

$$Y_{iq}^*(n)=W^\circ\, Y(t) \bigoplus_{R_i \in K} r_i \bigoplus_{R_j \notin K} r_j, \quad (4)$$

$$\rho(g_\Sigma, Y_{iq}^*(n))<\rho(g_i, Y_i^*(n)) \quad (5)$$

$$\rho(g_\Sigma, Y_{iq}^*(n))<\rho(g_q, Y_q^*(n)) \quad (6)$$

Выражения (5) и (6) означают, что возможность достижения цели роботы от объединения ресурсов больше, чем, если бы они действовали самостоятельно. Однако отметим, что получить реальные прогнозы применения своих стратегий роботы могут только в условиях, когда роботы, не принадлежащие коалиции, бездействуют, т.е. $\bigoplus_{R_j \notin K} r_j=0$ или агрегированные ресурсы коалиции доминируют ресурсы всех остальных роботов, т.е. $\bigoplus_{R_i \in K} r_i >> \bigoplus_{R_j \notin K} r_j$. Если роботы–противники обладают достаточным ресурсом и противодействуют роботам команды, то

формальная модель (4)(5)(6) не гарантирует достижения командой общей цели. В этих условиях используем эвристику: агент, имеющий большие возможности достижения цели, имеет большую привлекательность для командной работы. В нашей постановке задачи можно говорить, что возможность достижения цели у робота тем выше, чем ближе он находится к целевому объекту, т.е. к объекту или роботу, параметры которого нужно изменить до целевых значений. Можно задать окрестность близости роботов коалиции к целевому объекту ε_k, в которой роботы считаются привлекательными, т.е. $\rho(Y_i(t), Y_q(t))<\varepsilon_k$.

Тогда правило поведения роботов при функционировании команды, заключается в выборе и объединении ресурсов лучших роботов. Формально правило поведения каждого робота команды запишем так:

Если $\rho(Y_i(t), Y_q(t))<\varepsilon_k$, $\forall R_q \in K$.

То $Y_i^*(n)=W^\circ\ Y(t) \oplus r_i \bigoplus\limits_{R_j \notin K}^{N} r_j$

В результате независимых действий каждого роботы будет стихийно сформирована агрегированная стратегия лучших роботов команды, т.е.

$$Y_K^*(n)=W^\circ\ Y(t) \bigoplus_{R_i \in K} r_i \bigoplus_{R_j \notin K} r_j,$$

где стратегия команды $\bigoplus\limits_{R_i \in K} r_i$ включает стратегии роботов, имеющих большие возможности достижения цели из-за близости целевого объекта, т.е. $\rho(Y_i(t), Y_q(t))<\varepsilon_k$.

3.3 Принцип «ленивости» робота

При командной работе на основе принципа взаимной полезности, в условиях, когда неизвестны ресурсы групп роботов, не входящих в коалицию K, объединение ресурсов лучших роботов может быть избыточной. Рассмотрим принцип «ленивости» робота. Правило поведения каждого робота в этом случае будет выглядеть так:

Условие 1. **Если** $\rho(g_i, Y_i(t))>\rho(g_i, Y_i(t+1))$

$r_i^*=0$, $Y_i^*(n)=W^\circ\ Y(t) \oplus r_i^* \bigoplus\limits_{R_j \notin K}^{N} r_j$

То *переход на проверку условия 1*

Иначе

Условие 2. **Если** $\rho(Y_i(t), Y_q(t))<\varepsilon_k$, $\forall R_q \in K$.

То $Y_i^*(n)=W^\circ\ Y(t)\oplus r_i \overset{N}{\underset{R_j\notin K}{\oplus}} r_j$

переход на проверку условия 1

При выполнении действий, основанных на принципе «ленивости» робота, все роботы контролируют состояние среды (Условие 1). Если состояние среды меняется и целевой объект приближается к их цели, то действия отсутствуют. В противном случае роботы пытаются изменить состояние объекта в направлении цели. При этом свойства объекта изменяет команда из роботов, ресурсов, которых достаточно для изменения свойств объекта. Сам процесс образования такой «ленивой» команды происходит на основе только контроля состояния среды и не предполагает обмен информацией между роботами команды.

В командах роботов на основе полезности и «ленивости» работу по достижению цели будут выполнять лучшие роботы, в то время как роботы, имеющие меньшие возможности достижения цели будут «простаивать». Этот недостаток команды на основе принципа «ленивых» роботов можно устранить, если к принципам взаимной полезности и «ленивости» добавить еще принцип «эгоистичности» роботов.

3.4 Принцип «эгоистичности» робота

При разработке принципа «эгоистичности» робота рассматривается эвристика, основанная на том, что за совместную работу по достижению цели роботы получают вознаграждение. Вознаграждение это параметр, позволяющий регулировать функционирование команды. Считается, что все роботы команды, независимо от их возможностей по достижению цели стремятся получить вознаграждение. В командах, построенных на принципах полезности и «ленивости» вознаграждения будут получать только лучшие роботы. Поскольку роботы не способны обмениваться сообщениями, а могут только анализировать состояние среды функционирования, будем считать, что лучшие роботы, которых мы определили как роботов, находящихся ближе к целевому объекту будут получать большие вознаграждения за работу и, следовательно, иметь большую полезность. Т.е. $\rho(g, Y_i(t))>\rho(g, Y_j(t))\Rightarrow O(r_i)>O(r_j)$.

Этот принцип будем применять совместно с принципом «ленивости» роботов модифицировав условие 2:

Условие 2. **Если** $\rho(Y_i(t), Y_q(t))>\varepsilon_k, \forall R_q\in K$.

В этом случае в работе команды роботов будут принимать участие роботы, имеющие не лучшие возможности достижения цели, но они будут получать вознаграждения в случаях, если достичь цели все-таки удастся.

Второй вариант модификации принципа «ленивых» роботов заключается в исключении условия 2. Т.е.:

Иначе

$$Y_i^*(n)=W^\circ Y(t)\oplus r_i \bigoplus_{R_j \notin K}^{N} r_j$$

переход на проверку условия 1

В этом случае, роботы вне зависимости от их возможности достижения цели будут подключаться к выполнению общей работы на основе принципа «ленивости» робота, т.е. в случае выполнения условия 1. Такой алгоритм дает возможность роботу с любой возможностью достижения цели, достичь ее в команде и получить вознаграждение.

4 Требования к полноте информации о состоянии среды функционирования

Рассмотренные принципы формирования и функционирования стихийных команд роботов не предполагают, что каждый робот обладает полной информацией о состоянии среды функционирования $Y(t)$. Для выполнения своей задачи достижения цели роботу достаточно знать текущие $Y(t)$ и целевые $Y^*(t)$ значения целевых объектов или роботов. Будем считать, что неопределенность возможна как в целевых значениях параметров робота, так в текущих состояниях целевых объектов.

Пусть параметры робота или объекта A, $Y_A(t)= (z_{1j}, \ldots, z_{nb})$, $z_{ij} \in Z_{Ai}$. Неопределенность в векторе состояния будем обозначать знаком вопроса «?». Например, в векторе параметров $Y_A(t)= (z_{1j}, ?, z_{3j}, ?, \ldots, z_{nb})$ неопределенны второй и четвертый параметры, т.е. z_{2A} и z_{4A}. Поведение робота зависит от неопределенности в параметрах целевого объекта и в целевых параметрах и от характера задачи, решаемой роботом.

Для задачи фуражировки условия командной работы следующие: 1) для изменения свойств целевых $Y_A(t)$ объекта A, робот B должен изменить свои свойства $Y_B(t)$ так, чтобы взаимодействие было возможно; 2) робот B может изменить свойства объекта A до целевых $Y^*(t)$ при условии достаточности ресурсов. В задаче фуражировки эти условия означают, что робот B должен подойти к целевому объекту A и только после этого изменить его параметры самостоятельно или в команде.

Рассмотрим поведение робота в зависимости от различного сочетания неопределенностей:

- Неопределенностей нет. Известны текущие параметры целевого объекта и его целевое состояние.
- Неопределенность в параметрах целевого объекта.
- Неопределенность в параметрах целевого объекта и в параметрах цели.

Рассмотрим предложенные принципы и правила поведения в условиях неопределенности на простых примерах.

5 Пример и эксперименты

Для исследования формирования и функционирования стихийных команд роботов рассмотрим задачу фуражировки стихийными командами роботов. Вектор состояния среды функционирования $Y(t)=(Y_1(t),\ldots, Y_n(t))$ – это вектор параметров робота или объекта – $Y_i = (x_i, y_i, c_i, w_i)$, где x_i, y_i – координаты на плоскости, c_i - цвет робота и w_i – его вес. Робот характеризуется четверкой: цель G_i, ресурсы R_i, возможность самостоятельного достижения цели $\mu(Y(n),G_i)$ и полезностью достижения цели $O(R_i)$. Цель формулируется как вектор: $G_i=(G_{i1},\ldots, G_{in})$, где G_{ij} – вектор целевых параметров *j-го* целевого объекта, которые должен достичь *i-й* робот. Ресурсы R_i связаны с весом w_i робота. Возможность самостоятельного достижения цели $\mu(Y(n),G_i)$ представляется как функция от параметров самого робота или других роботов или объектов. При рассмотрении принципов организации стихийных команд возможность достижения цели была представлена как эвристическая функция расстояния до целевого объекта, т.е. $\rho(g, Y_i(t))$. Аналогично, как эвристика была определена функция полезность достижения цели $O(R_i)$.

Далее будем считать, что роботы могут менять только координаты целевых объектов. Изменения веса w_i и цвета c_i недоступны. Условия фуражировки стихийной командой роботов - $\bigoplus_{R_i \in K} w_i > w_j$, т.е. вес целевого объекта меньше агрегированного веса стихийной команды. Рассмотрим возможные варианты поведения роботов, при разных неопределенностях в параметрах целевого объекта и параметрах цели.

- Параметры целевого объекта (x_i, y_i, c_i, w_i) и цели (x_i^*, y_i^*, c_i^*, w_i^*) полностью определены. В этом случае может возникнуть стихийная команда роботов, работающая по любому из рассмотренных принципов и переместить целевой объект в заданную точку.
- Есть неопределенность целевого объекта в параметрах целевого объекта (x_i, *?*, c_i, w_i), а цель определена точно - (x_i^*, y_i^*, c_i^*, w_i^*). В этом случае любая неопределенность в параметрах целевого объекта приводит к поиску роботами объекта для любых значений неопределенного параметра. В этом случае роботы, имеющие общие цели могут найти разные целевые объекты, для фуражировки которых у них может оказаться недостаточно ресурсов. В этом случае стихийная команда может не образоваться.

- Целевой объект определен точно (x_i, y_i, c_i, w_i), но есть неопределенность в параметрах цели (x_i^*, *?*, c_i^*, w_i^*). В этом случае, неопределенность в цели позволяет роботам изменять значение неопределенного параметра до любых значений. Стихийная команда может образоваться.

В задаче фуражировки полезность каждого робота для командной работы определяется его близостью к целевому объекту и весом. Информация о параметрах других роботов недоступна. В работе [Кулинич, 2012] рассмотрена модель функционирования стихийных команд роботов, играющих в виртуальный футбол, в которой привлекательность робота для совместной работы оценивалась на основе анализа состояний всех роботов с помощью линейной свертки.

Заключение

В работе рассмотрены принципы формирования и функционирования стихийных команд роботов, основанные на анализе каждым роботом состояния среды функционирования и не предполагающие обмен роботами информацией о своих целях и ресурсах. Рассмотрены варианты поведения роботов в условиях неопределенности в параметрах целевого объекта и цели. Проводятся анализ возможности образования стихийных команд роботов в условиях неопределенности.

Список литературы

[Reynolds, 1987] Reynolds, C.W. Flocks, Herds, and Schools: A Distributed Behavioural Model // Computer Graphics. – 1987. – Vol. 21, No. 4, – PP. 25–34.

[Бурдун и др., 2010] Бурдун И.Е., Бубин А.Р. Метод самоорганизации стайного поведения малых мобильных роботов гражданского и специального назначения для арктических приложений. Сб. трудов Всероссийской научно-технической конференции «Научное и техническое освоение шельфа Северного Ледовитого океана». СибГУТИ. Новосибирск. 2010. С. 141-149.

[Карпов и др., 2019] Карпов В.Э., Карпова И.П., Кулинич А.А. Социальные сообщества роботов. М.: URSS, ООО «ЛЕНАНД», 2019. – 352 с.

[Кулинич, 2016] Кулинич А.А. Модели стайного поведения роботов / Всероссийский научно-практический семинар «Беспилотные транспортные средства с элементами искусственного интеллекта» (БТС-ИИ-2016) , 22-23 сентября 2016, Казань-Иннополис, с.60-69.

[Кулинич, 2012] Кулинич А.А. Модель поддержки принятия решений для образования коалиций в условиях неопределенности // Искусственный интеллект и принятие решений. № 2, 2012, стр. 95-106.

УДК 007.52, 595.796

КАЧЕСТВЕННЫЕ И КОЛИЧЕСТВЕННЫЕ ХАРАКТЕРИСТИКИ БИОИНСПИРИРОВАННЫХ МОДЕЛЕЙ ГРУППОВОЙ РОБОТОТЕХНИКИ

Е.В. Бургов (burgov.ev@yandex.ru)
НИЦ «Курчатовский институт», Москва

А.А. Малышев (a.san.malyshev@mail.ru)
НИЦ «Курчатовский институт», Москва

Аннотация. В рамках работы рассматривается решение задачи автономного сбора ресурсов группой роботов. В качестве модельного объекта для создания искусственной системы фуражировки выбраны муравьи. Показаны важные для моделирования добычи ресурсов характеристики их рабочих особей и семей. Выделенные особенности переложены на техническую систему – коллектив роботов. Построен полигон, разработана модель кормового поведения агентов и их система связи.[1]

Ключевые слова: групповая робототехника, биоинспирированный подход, модели поведения, муравьи, фуражировка, сбор ресурсов

Введение

Среди многообразия подходов к решению задач групповой робототехники, особый интерес представляют биоинспирированные модели и методы.

Биоинспирированность подразумевает как создание технических аналогов морфологических структур животных, так и вопросы социальной организации [Карпов, 2018] и поведенческих аспектов. В качестве модельных объектов часто используются общественные насекомые – муравьи, пчёлы, термиты. Так, биоинспирированные роботы своей морфологией, актуаторами, архитектурой системы управления, другими характеристиками имеют сходство с живыми организмами [Floreano et al., 2000].

С другой стороны, существует большое количество алгоритмов, основанных на наблюдении за животными: муравьиный алгоритм (Ant

[1] Работа выполнена при частичном финансировании РФФИ (проект № 17-29-07083 офи_м).

Colony Optimization [Dorigo et al., 2004]), алгоритм пчелиной колонии (Bees algorithm [Karaboga et al., 2007]), алгоритм светлячка (firefly algorithm [Yang, 2010]) и другие. Немало проектов в которых воссоздаются отдельные природные механизмы: на основе алгоритма летучей мыши [Suárez et al., 2019] – поиск цели в закрытом неисследованном помещении; симуляция химической коммуникации у роботов (виртуальная – световой след на поверхности LCD-экрана [Arvin et al., 2015]; реальная – с использованием этанола [Fujisawa et al., 2014]); строительство искусственных сооружений [Petersen et al., 2011].

Однако, при проведении аналогий, на основе которых строятся методы, нередко опускаются существенные детали, важные для поведения животных, делаются допущения, которые могут быть подвергнуты серьёзной критике. Например, существует представление о ведущей роли химической коммуникации у муравьев, на котором основано широкое применение «муравьиного» алгоритма. Для некоторых видов муравьев это положение справедливо [Wilson, 1962], но у других доминируют иные коммуникационные системы [Федосеева, 2015]. А значит и применение «муравьиного» алгоритма должно быть с одной стороны ограничено, а с другой – дополнено использованием других систем навигации и коммуникации.

Рассмотрим одну из классических задач групповой робототехники – фуражировку, применив биоинспирированный подход. Исследуя биологическую систему, обратим внимание на её характеристики, механизмы, принципы, обеспечивающие её функционирование.

1 Фуражировка

Фуражировку будем рассматривать, как часть более крупной задачи по поддержанию энергетической автономности группы роботов. Таким образом, фуражировка производится с целью добычи ресурсов. Основные объекты фуражировки: группа роботов (агентов); ресурсы – собираемые объекты; источники ресурсов – места концентрации ресурсов; база – место переработки ресурсов; полигон – территория, на которой функционируют агенты. В работе рассматриваются только вопросы, касающиеся фуражировки.

Фуражировка включает в себя ряд задач: нахождение, сбор, выгрузка ресурсов, определение местоположения базы, информирование членов коллектива об обнаруженных источниках ресурсов и другие. Она производится на ограниченной территории группой роботов. Важную роль играет база – область, куда роботы привозят собранные ресурсы, где они делятся информацией о местоположении источников, мобилизуют других роботов для сбора.

Каждая задача требует конкретного технического решения. Обратившись к биологической системе – семье муравьёв, рассмотрим, как задача фуражировки решается насекомыми. В качестве модельного объекта выбраны муравьи, так как они характеризуются достижением высшего уровня сложности социальных структур в мире насекомых, успешно решают задачу групповой фуражировки.

2 Муравьи как модельный объект в групповой робототехнике

Муравьи являются уникальным модельным объектом, изучение которого может способствовать решению многих задач групповой робототехники. Они характеризуются достижением высшего уровня сложности социальных структур в мире насекомых. Муравьи эффективно решают такие задачи, связанные с фуражировкой, как:

– разведка, контроль, использование и охрана территории;
– добыча ресурсов, их обработка и распределение внутри группы;
– создание инфраструктуры, включающей различные элементы, способствующие эффективному использованию кормового участка.

Нередко исследователи обращаются к муравьиному алгоритму [Khaluf et al., 2019]. В его основе лежит принцип непрямого взаимодействия между индивидами посредством изменения окружающей среды (стигмергии) [Dorigo et al., 2004] и, конкретнее, использования некоторыми видами муравьев химического следа (феромонов) для формирования дорог и мобилизации [Wilson, 1962]. Фуражиры муравьев *Solenopsis*, *Lasius* и других родов, возвращаясь в гнездо после обнаружения источника пищи, оставляют пахучие метки по пути своего следования. Далее, при интенсивном использовании формирующейся дороги, количество химических меток увеличивается, и помеченный маршрут становится все более привлекательным для муравьев.

Нередко этот механизм рассматривается как основной в процессе самоорганизации сообществ насекомых. Однако его изолированное использование, как при описании функционирования семей муравьев, так и при моделировании, практически невозможно по ряду причин. Во-первых, не все муравьи так широко используют химический след. Во-вторых, даже при использовании муравьями феромонного следа, остается актуальной разведка, во время которой у индивида так же должна быть система ориентирования в пространстве. И при попытке использовать механизм аналогичный химическому следу для оптимизации работы группы роботов непременно возникают сложности первичного освоения территории, когда приходится ориентироваться на местности без «разметки».

Многие исследования показывают, что при организации групповых действий у муравьев комбинируются различные механизмы ориентации и коммуникации. Например, при организации рейдов муравьи-рабовладельцы *Polyergus* ориентируются по наземным объектам, по поляризации света и с использованием химического следа [Mori et al., 2001]. На этом и других примерах видно, что при проектировании группы роботов, необходимо использовать не отдельно взятые механизмы, которые работают в природе, а целые (насколько это возможно) системы.

Для моделирования важными параметрами являются структура социума и количество индивидов в нём. Социум муравьев по своему устройству и происхождению является семьей. Описано три уровня внутрисемейных структур муравьёв: клан, колонна и плеяда. Первый уровень – клан – объединение рабочих, основанное на индивидуальном взаимодействии, имеющее иерархическую структуру [Захаров, 1991]. Колонна и плеяда являются более сложными уровнями организации. Для решения задачи фуражировки достаточно реализации клана. Численность особей в семье варьируется в зависимости от вида, жизненной стадии группы, ее состояния. Численность семьи-клана – от нескольких десятков до нескольких сотен особей [Захаров, 2015]. В рамках работы она полагается равной 150-300 рабочих, так как семьи такой численности часто используются в лабораторных и полевых экспериментах [Длусский, 1981; Богатырева и др., 1998; Федосеева, 2015].

В семье постоянно происходит распределение и смена функций рабочих. Эти функции: забота о потомстве и самке, строительство и ремонт гнезда, добыча пищи. Особи, занимающиеся добычей пищи – фуражиры. Их численность поддерживается на уровне 10-15% от населения муравейника [Захаров, 2015]. Т.е. для семьи-клана из 150-300 особей их число составит 15-45. Кроме того, среди фуражиров происходит дополнительное функциональное разделение. Две основные группы фуражиров: активные (самостоятельно осуществляют поиск пищи) и пассивные (выполняют добычу пищи после активации – взаимодействия с активным фуражиром) [Захаров, 1972].

В качестве модельного объекта используются муравьи рода *Formica*, так как они отличаются способностью к формированию поселений разного масштаба и сложности [Бургов, 2016], способны эффективно использовать большие территории, уменьшать активность и численность других муравьев на них [Захаров, 2015]. Размеры рабочих *Formica* различны: 0,45-0,75 см – *Formica exsecta*, 0,4-0,7 см – *F. cunicularia*, 0,45-0,95 см – *F. pratensis* [Collingwood, 1979]. В недавнем эксперименте [Бургов, 2018] были получены значения скорости движения для этих видов муравьев. При движении по мостику шириной 0,4 см средние значения составили: 3 см/с у *Formica exsecta*, 3,6 см/с у *F. cunicularia*, 4,5 см/с у *F. pratensis*.

Усреднённые значения, соответственно, составляют 0,62 см для размера тела и 3,7 см/с для скорости. Дальнейшие расчёты производятся на основе этих значений.

У муравьев есть ряд способов передачи информации друг другу: тактильный код, кинопсис (язык поз), химический след, звуковые сигналы («стрекотание») [Длусский, 1981]. Для муравьев *Formica* особенно важны первые два механизма, поэтому необходимо моделировать системы ближней (информационной) связи, и дальней (сигнальной). Передача информации у муравьев с использованием тактильного кода происходит при непосредственном контакте особей, а посредством языка поз – на дистанции до ≈ 20 см (≈ 32 длины тела особи).

Практика показывает, что для длительного содержания в лабораторных условиях семей *Formica* численностью до 500 рабочих вполне достаточно двух арен размерами 80х40 см каждая. Значения эти очень грубые, однако их точности достаточно для начального моделирования.

При общем вторичном делении территории у многих *Formica* кормовой участок семьи разделяется на индивидуальные поисковые участки (ИПУ). ИПУ – фрагмент, обследуемый и используемый одним фуражиром [Захаров, 2015]. Оценить размеры ИПУ сложно, так как они зависят от количества ресурсов на территории, особенностей микрорельефа и т.д.

3 Экспериментальная база

Опираясь на данные о модельных сообществах муравьев, рассчитаем параметры: численность группы роботов, дистанцию их коммуникации и размер полигона («кормовой участок»).

Согласно оценке из предыдущего раздела, семье муравьев из 500 особей достаточно территории с суммарной площадью $S_{ку}$ = 6400 см^2. Введём понятие единичной площади S` – площадь, занимаемая одним объектом. Для муравья $S`_м = 0.62^2$ = 0,3844 см^2. Тогда площадь кормового участка муравьёв, выраженная через $S`_м$, составляет $S_{ку} \approx 16830$ ед.. Учитывая, что в семье муравьёв в среднем 13% фуражиров, для группы из 500 особей, их количество составит 65, на каждого из которых приходится s ≈ 260 ед. площади кормового участка.

При оценке размера площади стоит учитывать и то, как быстро муравьи перемещаются. В качестве величины перемещения индивидов используем относительную скорость – расстояние, выраженное в линейных размерах объекта, преодолеваемое им за 1 с. Для муравья она составляет ≈ 6 1/с.

Экспериментальным базисом является гетерогенная группировка, состоящая из 14 роботов: 10 роботов YARP-2 и 4 робота Dr.YARP (Рис. 1). Её численность соответствует минимальному количеству фуражиров в семье-клане, поэтому использование допустимо.

Рис. 1. Группа роботов YARP-2 и Dr.YARP

На YARP-2 сенсорная система представлена пятью ИК-дальномерами дальностью до 80 см. Робот оснащён системой локальной связи (маяком). Размеры (ДхШхВ): 21х21х23 см, скорость – до 10 см/с.

Dr. YARP оснащён четырьмя УЗД датчиками, пятью ИК-дальномерами дальностью до 80 см и камерой. На борту также установлен маяк. Размеры Dr.YARP: 40х30х25 см, скорость – до 20 см/с.

Маяк – программно-аппаратный комплекс для передачи данных в ИК-диапазоне. Передаваться могут как команды из ограниченного набора (сигнальная связь, передача 3 битовой команды на расстояние до 270 см), так и информация (информационная связь, передача 12 бит информации на расстояние до 100-150 см).

Для экспериментальной группы роботов $S_{ку} = s*(10* S`_{yarp2} + 4* S`_{dryarp}) = 260 * (10*441 + 4*1200) = 2394600$ см$^2 \approx 240$ м^2, относительная скорость $\approx 0{,}5$ 1/с.

Разработанный полигон (Рис. 2) имеет размер 480х900 м^2 (43,2 м^2), по периметру ограничен невысоким ограждением. Для определения положения роботов (координаты и угол поворота относительно некоторой оси), полигон оснащён системой локализации, аппаратная часть которой включает 6 ip-камер, объединённых в одну локальную сеть.

В рамках задачи фуражировки к базе не предъявляются функциональные требования. Поэтому в качестве базовой станции достаточно использовать часть полигона, каким-либо образом определённую как база, например, маяком и разметкой.

Источники ресурсов – области полигона, оснащённые маяками, где располагаются ресурсы. Каждый маяк излучает сигнал о том, какой ресурс находится в этой области.

Рис. 2. Полигон. Пунктиром выделены IP-камеры.

Полигон и группа роботов близка по целевым характеристикам к условной семье-клану муравьёв *Formica* (Таблица 1). Указанные характеристики обоснованы в разделе 2.

Таблица 1. Сравнение характеристик модельной семьи-клана муравьев *Formica* и группы роботов.

Параметр	Модельная семья-клан муравьев *Formica*		Модельная группа роботов
Численность группы, ед.	Общая	Фуражиры	14
	150-300	15-45	
Доля кормового участка на одного фуражира, ед	260		~ 47
Территория модельной группы, $м^2$	0,64		43,2
Относительная скорость, 1/с	6		0,5
Макс. дистанция локального взаимодействия (информационная обмен), ед. линейных размеров	0 (непосредственный тактильный контакт)		4
Макс. дистанция передачи коротких сигналов (сигнальная связь), дистанция / линейный размер объекта	~ 32		13

4 Модель кормового поведения

В модели поведения определяются следующие основные объекты:

– гнездо, базовая станция – место базирования роботов;
– источники ресурсов – области, где возможна добыча ресурсов;
– ресурсы – объекты от сбора которых зависит функционирование группы. В общем случае есть несколько разновидностей ресурсов;
– фуражир – индивид, занятый добычей пищи.

Модель поведения предполагает наличие активных и пассивных фуражиров. Активный фуражир (АФ) – фуражир, самостоятельно осуществляющий поиск пищи. Пассивный фуражир (ПФ) – индивид, фуражировочная деятельность которого инициируется АФ.

Фуражировка – комплексная задача. Её можно разбить на несколько этапов: 1) Поиск ресурсов; 2) Добыча ресурсов; 3) Возвращение на базу; 4) Транспортировка ресурсов на базу; 5) Информирование других агентов о местонахождении ресурсов.

Поведение каждого робота задаётся мета-автоматом, который осуществляет переключение состояний, представленных автоматами фиксированного комплекса действий (ФКД). ФКД состоит из базовых поведенческих процедур. Все автоматы имеют форму автоматов Мили.

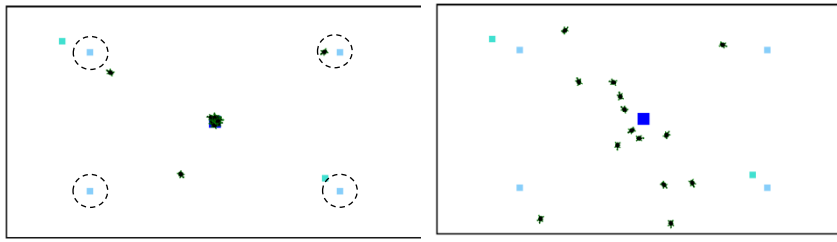
В начале моделирования часть индивидов являются АФ. Если пища не обнаружена к моменту, когда роботу требуется подзарядка, АФ возвращаются в гнездо и становятся ПФ. АФ, обнаружившие пищу, возвращаются в гнездо и инициируют фуражировку ПФ.

Поскольку размер группы роботов ограничен, модель предполагает возможность быстрой смены статуса фуражира с пассивного на активный и обратно. Последний переход в природе маловероятен, но при моделировании допустим. Допущение о возможности таких переходов позволяет малой группе роботов увеличить мобилизационный потенциал.

5 Эксперименты

В рамках экспериментов отработаны элементы модели кормового поведения: 1) Поиск ресурсов, базовой станции. 2) Неспецифическая активация пассивных фуражиров. Моделирование поведения проводилось в среде kvorum (Рис. 3).

Базовая станция расположена в центре карты. При инициализации, все агенты находятся на ней. На карте имеется 6 источников ресурсов – 4 пригодны для питания агентов, 2 других – нет (Рис. а). На первом шаге количество активных фуражиров небольшое – 3-4 особи, остальные пассивные. Когда агент возвращается на базу с ресурсом, он активирует пассивных фуражиров (Рис. б).



а) б)

Рис. 3. Моделирование кормового поведения: а) шаг 1 – 3 активных фуражира, выделены пригодные источники; б) после активации – все агенты ищут ресурсы

Заключение

В работе рассмотрено решение классической задачи групповой робототехники – фуражировки, биоинспирированным подходом. В качестве модельного объекта была выбрана семья муравьёв *Formica*. Рассмотрены основные параметры семьи, существенные для фуражировки, на их основе определены размеры полигона, численность роботов и дистанция их взаимодействия – информационного и сигнального, описана модель поведения роботов. В рамках экспериментов в среде моделирования kvorum были отработаны её элементы.

Список литературы

[Arvin et al., 2015] Arvin F., Krajnik T, Turgut AE, Yue S. COSΦ: Artificial pheromone system for robotic swarms research // IEEE Int. Conf. Intell. Robot. Syst. 2015. с. 407–412.

[Collingwood, 1979] Collingwood C.A. The Formicidae (Hymenoptera) of Fennoscandia and Denmark. Klampenborg, Denmark: Scandinavian Science Press Ltd., 1979. 156 с.

[Floreano et al., 2000] Floreano D., Mattiussi C., Brooks R. Bio-Inspired Artificial Intelligence: theories, methods, and technologies. London: The MIT Press, 2000.

[Fujisawa et al., 2014] Fujisawa R., Dobata S, Sugawara K, Matsuno F. Designing pheromone communication in swarm robotics: Group foraging behavior mediated by chemical substance // Swarm Intell. 2014. Т. 8. № 3. с. 227–246.

[Karaboga et al., 2007] Karaboga D., Basturk B. A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm // J. Glob. Optim. 2007. Т. 39. № 3. с. 459–471.

[Khaluf et al., 2019] Khaluf Y., Vanhee S., Simoens P. Local ant system for allocating robot swarms to time-constrained tasks // J. Comput. Sci. 2019. Т. 31. с. 33–44.

[Mori et al., 2001] Mori A., Grasso DA, Visicchio R, Le Moli F. Comparison of reproductive strategies and raiding behavior in facultative and obligatory slave-

making ants: the case of *Formica sanguinea* and *Polyergus rufescens* // Insectes Soc. 2001. Т. 48. № 4. с. 302–314.

[Petersen et al., 2011] Petersen K., Nagpal R., Werfel J. TERMES: An Autonomous Robotic System for Three-Dimensional Collective Construction // Robotics: science and systems. Los Angeles: , 2011.

[Suárez et al., 2019] Suárez P., Iglesias A., Gálvez A. Make robots be bats: specializing robotic swarms to the Bat algorithm // Swarm Evol. Comput. 2019. Т. 44. с. 113–129.

[Wilson, 1962] Wilson E.O. Chemical communication among workers of the fire ant *Solenopsis saevissima*. 1. The organization of mass foraging; 2. An information analysis of the odor trail; the experimental induction of social responses // Anim. Behav. 1962. Т. 10. с. 134–147, 148–158, 159– 164.

[Yang, 2010] Yang X.-S. Nature-Inspired Metaheuristic Algorithms Second Edition. , 2010. 115 с.

[Богатырева и др., 1998] Богатырева О.А., Шиллеров А.Е. Синергетика социальности. Новосибирск: Изд-во СО РАН: , 1998. 292 с. с.

[Бургов, 2016] Бургов Е.В. Пространственно-функциональные структуры у муравьев *Serviformica* (Hymenoptera: Formicidae) // Вестник МГПУ. Серия «Естественные науки». 2016. Т. 24. № 4. с. 19–27.

[Бургов, 2018] Бургов Е.В. Функциональные основы экологической сегрегации видов у муравьев: предварительные данные // Муравьи и защита леса (Материалы XV Всероссийского мирмекологического симпозиума 20-24 августа 2018 года). , 2018. с. 25–31.

[Длусский, 1981] Длусский Г.М. Принципы коммуникации у муравьев // Доклады на 33-м ежегодном чтении памяти Н.А. Холодковского. Л.: Наука, 1981. с. 3–33.

[Захаров, 1972] Захаров А.А. Внутривидовые отношения у муравьев. М.: Наука: , 1972. 216 с.

[Захаров, 1991] Захаров А.А. Организация сообществ у муравьев. М.: Наука: , 1991. 278 с.

[Захаров, 2015] Захаров А.А. Муравьи лесных сообществ, их жизнь и роль в лесу. М.: Товарищество научных изданий КМК, 2015. 404 с.

[Карпов, 2018] Карпов В.Э., Карпова И.П., Кулинич А.А. Социальные сообщества роботов. Москва: ЛЕНАНД, 2018.

[Федосеева, 2015] Федосеева Е.Б. Технологический подход к описанию групповой фуражировки муравьев *Myrmica rubra* // Зоол. журн. 2015. Т. 94. № 10. с. 1163–1178.

УДК 004.932.2

АРХИТЕКТУРА ПРОГРАММНО-АППАРАТНОГО КОМПЛЕКСА БЕСПИЛОТНОГО АВТОМОБИЛЯ ДЛЯ КОНКУРСА «ЗИМНИЙ ГОРОД»

А.К. Буйвал (*a.buyval@innopolis.ru*)
А.Р. Габдуллин (*a.gabdullin@innopolis.ru*)
Р.В. Федоренко (*r.fedorenko@innopolis.ru)*
М. С. Любимов (*m.liubimov@innopolis.ru)*

Университет Иннополис, Иннополис

Аннотация. В данной статье рассматривается архитектура программно-аппаратного комплекса используемого командой Университета Иннополис для конкурса "Зимний город". Также в статье приводятся основные технические решения и их обоснования для каждой из ключевых подсистем программного комплекса: локализация, планирование маршрута и управление, распознавание объектов и прогнозирование их поведения. В статье приведены результаты тестирования разработанного программного комплекса в различных ситуациях на электромобиле КИА.[1]

Ключевые слова: беспилотный автомобиль, конкурс «Зимний город», локализация, обнаружение объектов на изображении.

Введение

Технологические конкурсы за последние десятилетия показали свою эффективность для решения сложных задач, требующих привлечения большого количество интеллектуальных и финансовых ресурсов. В качестве примера можно привести конкурс DARPA Grand Challenge [DARPA, 2004], который стал толчком для развития не только беспилотных автомобилей, но и их различных компонентов. В 2018 году правительство РФ объявило конкурс «Зимний город» направленный на преодоление технологического барьера – создание беспилотного автомобиля способного

[1] Эта работа была поддержана Министерством науки и высшего образования Российской Федерации, в рамках проекта «Разработка модульной системы дистанционного и автономного управления коммерческим транспортом совместно с комплексом аэроразведки маршрута движения на базе отечественных компонентов» (Соглашение о предоставлении субсидии № 075-10-2018-011 (№14.609.21.0100), Уникальный Идентификатор RFMEFI60917X0100)

функционировать в сложных погодных условиях. Основный критерий конкурса заключается в том, что беспилотный автомобиль должен преодолеть 50 км за 3 часа в сложных дорожных условиях. В данной статье мы представляем архитектуру программного-аппаратного комплекса беспилотного автомобиля, которую мы, как команда Университета Иннополис, используем для решения задач конкурса.

1 Архитектура аппаратной части

В качестве базы для беспилотного автомобиля мы выбрали электромобиль KIA Soul 2014. Выбор данной модели был обусловлен следующими факторами:

- емкий аккумулятор электромобиля позволяет обеспечить стабильной электроэнергией вычислительный модуль;
- возможность управление рулевой колонкой путем имитации данных с датчика крутящего момента электроусилителя руля;
- возможность управление тягой и торможением путем имитации данных с педалей газа и тормоза.

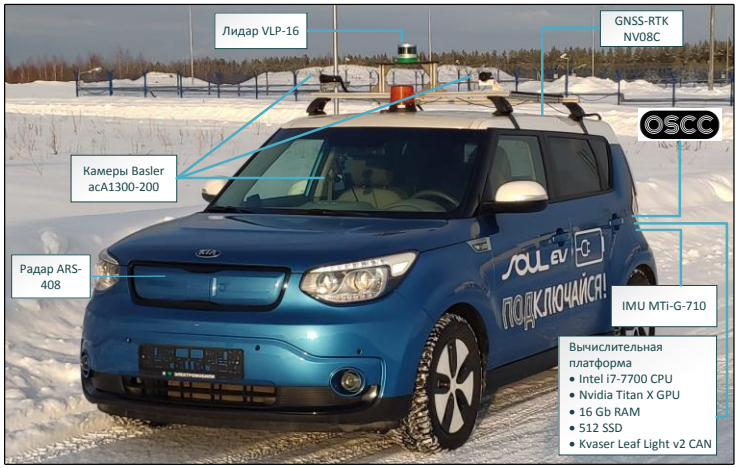


Рис. 1. Беспилотный автомобиль со схемой установленного оборудования.

Для передачи сигналов управления от вычислительного модуля к агрегатам автомобиля был использован набор модулей на базе план Arduino из открытого проекта Open Source Car Control [OSCC, 2019].

Также на автомобиль были установлены следующие сенсоры:

- 16 лучевой лазерный сканер (лидар) Velodyne VLP-16;
- 3 видеокамеры Basler AC A1300-200;

- датчик глобального позиционирования NV08C-RTK;
- датчик инерциальной системы навигации, включающий гироскоп, акселерометр и компас Xsense MTi-G-710;
- радар Continental ARS-408.

На рис. 1 представлена фотография беспилотного автомобиля с установленными датчиками.

В качестве бортового вычислительного модуля мы использовали персональный компьютер на базе процессора Intel Core i7 и с видеоадаптером Nvidia Titan X с 12 ГБ графической памяти, что позволило использовать сверточные нейронные сети для быстрой детекции объектов.

2 Архитектура программной части

В качестве базы разрабатываемой системы управления беспилотным автомобилем мы использовали открытый проект «Apollo». Данный проект использует модифицированную версию операционной системы роботов ROS, которая позволяет передавать сообщения между модулями через оперативную память, что критично для сообщений с большим количество данных. Во многом общая архитектура система представленная на рис. 2, заимствована из проекта Apollo и типична для беспилотных автомобилей. Apollo взят в качестве основы по причине наличия развитого интерфейса пользователя на основе веб-технологий, необходимых библиотек и улучшенных средств межпроцессного взаимодействия на основе protobuf, а также совместимости с симулятором беспилотных автомобилей LGSVL Simulator.

Рис. 2. Общая архитектура программной части.

2.1 Модуль локализации

Источниками данных для работы системы локализации являются:

- система глобальной спутниковой навигации GPS/ГЛОНАСС;
- кинематика реального времени (RTK);
- колесная одометрия;

– датчик инерциальной навигации;
– сенсорная система автомобиля (лазерный сканер и камеры).

Каждое из этих устройств имеет собственные условия, в которых они эффективны, но ни одно из них по отдельности, как правило, недостаточно для управления автомобилем. Так, система спутниковой навигации имеет погрешность порядка 6 метров и не работает в туннелях и в условиях плотной застройки, система RTK позволяет улучшить точность работы системы GPS да нескольких сантиметров, но вносит дополнительные ограничения, связанные с получением поправок по радиоканалу. Частота данных (до 4 Гц), получаемых от системы GPS, также недостаточна для задач управления автомобилем. При этом система инерциальной навигации и колесная одометрия могут предоставлять данные с частотой порядка 100 Гц, однако рассчитанные с их помощью координаты содержат накапливающуюся со временем ошибку. Структура модуля локализации показана на рис. 3.

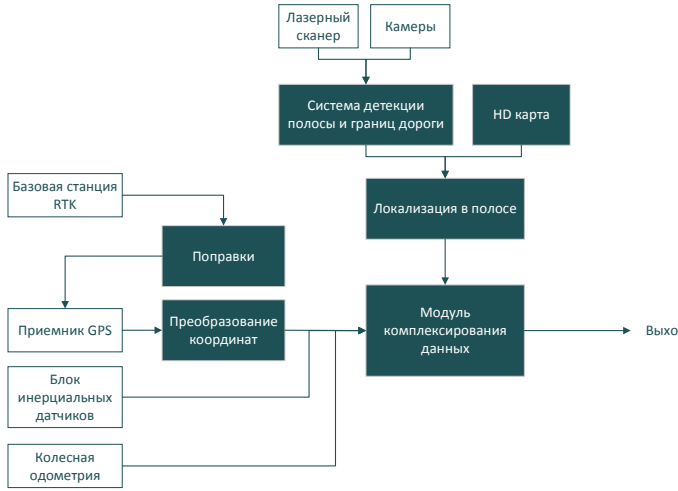


Рис. 3. Структура модуля локализации.

Таким образом, центральным модулем в системе локализации является модуль комплексирования данных различных устройств. Эта задача решается с помощью общепризнанных алгоритмов на основе фильтра Калмана [Wan et al., 2017].

Особенностью нашей реализации системы локализации автомобиля является возможность работы без системы RTK за счет использования HD-карты и системы детекции полосы и границ дороги для определения

ориентации и бокового смещения автомобиля относительно полосы на базе данных сенсорной системы (лазерного сканера и камер) без необходимости предварительного создания отдельных карт для системы локализации.

2.2 Подсистема обнаружения объектов

Общая структура модуля обнаружения объектов представлена на рис. 4.

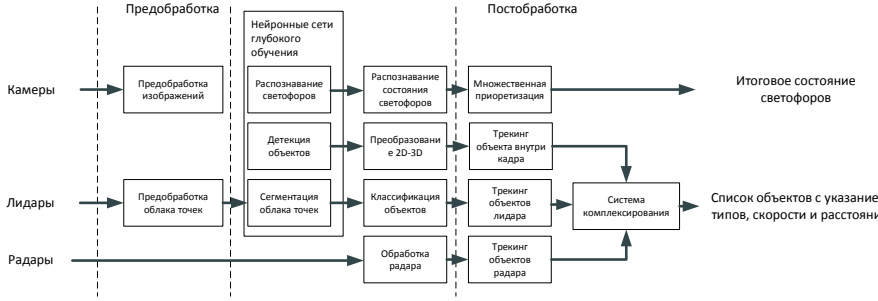


Рис. 4. Структура подсистемы обнаружения объектов.

Источниками сенсорной информации для модуля обнаружения объектов являются видеокамеры, лазерные сканеры (лидары) и радары. В состав модуля входят модуль обнаружения объектов в облаке точек, модуль обнаружения объектов по изображению с камеры, которые описаны далее, модуль комплексирования данных на основе фильтра Калмана, а также модуль распознавания состояния светофора.

2.2.1 Модуль обнаружения объектов в облаке точек

Существуют различные подходы для обнаружения объектов в облаке точек, генерируемых лидарами. Например, в работе [Li, 2016] предлагается использовать 3D сверточную нейронную сеть (СНС) для обнаружения объектов. Такой подход позволяет подавать на вход детектора облако точек практически в неизменном виде, однако такая СНС имеет большую вычислительную сложность и сильно зависит от модели лидара. В своей предыдущей работе [Buyval, 2018] мы использовали алгоритмы кластеризации на основе эвклидова расстояния для того, чтобы обнаружить кластеры и затем классифицировали их с помощью деревья решений. Однако такой подход плохо показал себя при большом количестве близко расположенных объектов.

В данной работе мы решили использовать СНС, но заранее трансформировать 3D облако точек в 2D поле признаков. Данное поле признаков представляет собой сетку ячеек в локальных координатах X-Y с

определенным шагом. Каждая ячейка имеет следующий набор признаков (каналов):

- максимальная и средняя высота точек в ячейки;
- количество точек в ячейки;
- средняя интенсивность отражения в ячейки;
- угол и расстояние от центра СК до ячейки;
- бинарное значение описывающие свободна или занята ячейка.

Далее мы подаем полученные значение на вход полносвязной СНС, которая выполняет задачу сегментации на данном поле признаков. На выходе мы получаем значения, характеризующие наличие объекта, его класс и его характеристики в каждой ячейке.

2.2.2 Модуль обнаружения объектов по изображению с камеры

В последние годы наиболее эффективными методами обнаружения объектов являются методы, основанные на использования глубинных сверточных нейронных сетей. В целом ряде исследований и задач показана высокая эффективность данных методов, однако узким местом для их использования остается высокая требовательность к вычислительным ресурсам. В нашем исследовании использовали сверточную нейронную сеть, основанную на архитектуре YOLOv3 [Redmon, 2018], т.к. она обеспечивает наилучшее соотношение точность/скорость среди существующих архитектур.

Помимо определения класса объекта и его положения на изображении важно также определить геометрические размеры объекта, а также его ориентацию на изображении. Зная эти характеристики объекта, мы затем сможем восстановить положение ограничивающего параллелепипеда в системе координат карты и передать эти данные системы трекинга.

Для того, чтобы определять геометрические размеры объекта мы добавили 3 дополнительные размерности в каждой из выходных сверточных слоев. Каждая дополнительная размерность отвечает за регрессию отклонения каждого геометрического размера от среднего размера определенного для каждого класса.

Для определения ориентации объекта мы добавили в выходные слои дополнительные 8 классов, представляющих грубую оценку ориентации, а также дополнительную размерность для коррекции угла каждого класса.

В работе [Mousavian, 2017] было акцентировано внимание на то, что визуально ориентация машина зависит не только от физического расположения автомобиля в пространстве, но также относительного положения машины от центра изображения. Для решения этой проблемы мы определяем видимую ориентацию объекта с помощью СНС, а затем, чтобы получить реальную ориентацию объекта добавляем коррекцию, основанную на положении объекта в кадре относительно центра кадра.

Таким образом окончательная ориентация объекта рассчитывается по следующей формуле:

$$\theta = \theta_{cl} + \theta_{corr} + \theta_{ray}$$

где θ_{cl} – ориентация соответствующая классу ориентацию с максимальной уверенностью на выходе СНС, θ_{corr} – коррекция ориентации для соответствующего класса, θ_{ray} – коррекция ориентации в соответствии с относительным положением объекта на кадре.

Значение θ_{ray} определяется по следующей формуле:

$$\theta_{ray} = \text{atan}(\frac{cx}{F})$$

где cx – отклонение центра объекта от центра кадра в пикселях, F – фокусное расстояние камеры в пикселях.

Финальная структура выходного слоя нашей сверточной нейронной сети представлена на рис. 5.

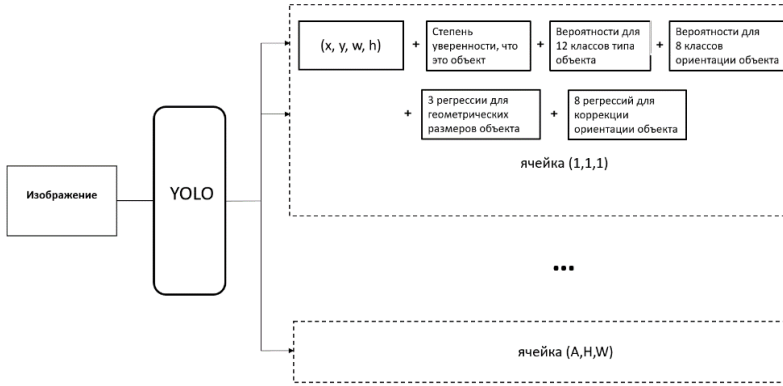


Рис. 5. Структура выходного слоя СНС.

Для обучения СНС мы использовали набор данных KITTI, как один из немногих наборов данных, который содержит данные об ориентации объекта и его геометрических размеров. Так как набор данных KITTI содержит только изображения в летний период с хорошей освещенностью, мы дополнили его собственным набором, собранным на территории города Иннополис в зимний период в дневное и ночное время.

На основе полученных с помощью СНС 2D координат, физических размеров объекта и его ориентации в пространстве далее мы делаем преобразование в 3D координаты в системе координат карты. На рис. 6 представлен пример обнаружения нескольких объектов в 3D координатах.

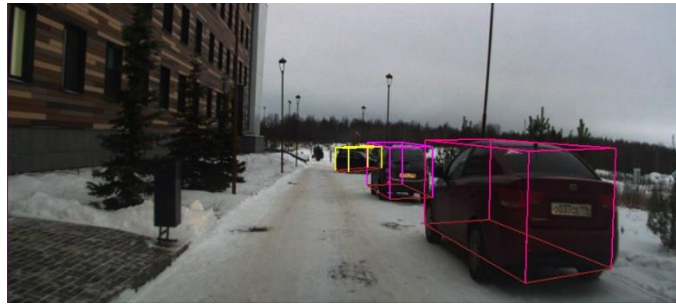


Рис. 6. Пример обнаружения объектов в 3D координатах.

2.2.3 Модуль распознавания состояния светофора

Данный модуль в качестве входных данных получает координаты обнаруженных СНС светофоров в системе координат изображения. На основе заранее известных координат светофоров в мировой системе координат (СК) мы фильтруем найденные на изображении светофоры таким образом, чтобы оставить только те, которые относятся к нашему маршруту движения. Затем изображение светофора передается на СНС, осуществляющую классификацию состояния светофора. Полученный результат затем обрабатывается с учетом известной логики переключения светофора, чтобы исключить ложные переключения.

2.3 Модули планирования траектории и управления

На момент написания данной статьи мы использовали модуль планирования из проекта Apollo. Он состоит из нескольких последовательных этапов:

- на основе базовой траектории, полученной из заданного пользователем маршрута и с учетом расположения препятствий, строится граф с минимальной стоимостью движения;
- через точки графа строится сплайн, также с минимальной стоимостью пути;
- для построенного сплайна строится профиль скорости по нему с учетом заданных ограничений на скорость и ускорения.

Полученная траектория с профилем скорости передается модулю управления. На текущий момент мы также использовали стандартный модуль управления проекта Apollo, который использует 2 независимых регулятора для управления продольной скоростью и для управления боковым отклонением от траектории. В качестве регулятора продольной скорости используется ПИД регулятор с калибровочной таблицей для

преобразования требуемого ускорения в значения нажатия педалей газа и тормоза. Регулятор бокового отклонения построен по принципу LQR регулятора и использует внутри простую динамическую модель автомобиля с 2-мя колесами.

В ходе экспериментов мы столкнулись с рядом сложностей в работе данных модулей. Наиболее серьезной проблемой было то, что модуль планирования часто выдавал абсолютно неприемлемую траекторию. В основном это было связано с тем, что система построения сплайна не могла найти оптимальное решение. Также нас не устроило то, что время планирования траектории было от 100 до 300 мс в зависимости от ситуации. В связи с этим мы решили заменить оба модуля на единый построенный по принципу Model Predictive Control. Использование подобного подхода уже показало свою эффективность в ряде наших исследований, в частности в [Buyval, 2017].

2.4 Человеко-машинный интерфейс

Для взаимодействия оператора автомобиля с системой управления используется веб-приложение (Dreamview) построенное на базе фреймворка react [React, 2019]. Оно реализовано как веб-приложение, запускаемое в браузере.

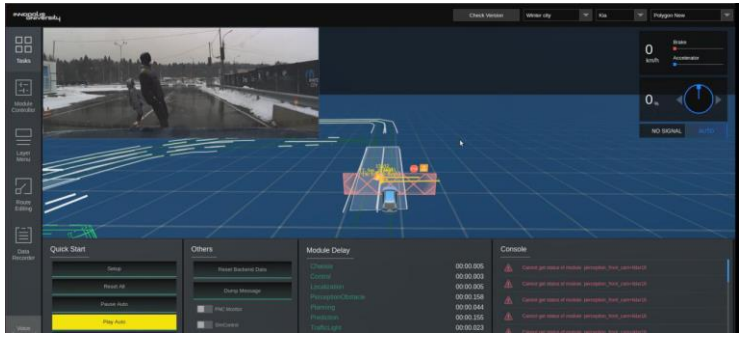


Рис. 7. Web-интерфейс программной системы.

Таким образом оно доступно для оператора как в автомобиле, так и вне его. Серверная часть данного приложения взаимодействует с системой управления автомобилем через каналы связи ROS/protobuf. Интерфейс позволяет отображать состояние запущенных программных модулей, позицию и ориентацию автомобиля на карте, препятствия и их прогнозируемые траектории в системе координат мира.

На рис. 7 показан интерфейс Dreamview в режиме автономного движения. В верхнем левом углу можно видеть изображение с курсовой

камеры. Планируемая траектория движения изображена кривой голубого цвета. Состояние распознанного курсового светофора изображено в верхнем правом углу.

3 Заключение

Конкурс «Зимний город» предоставляет возможности испытаний беспилотных автомобилей на полигоне, приближенном к городским условиям. Участие в таком конкурсе позволяет проверить на практике описанные в данной статье алгоритмы и подходы, найти их недостатки и способы улучшения.

Испытания представленной системы на полигоне показало, что автомобиль способен двигаться в автономном режиме даже в непростых погодных условиях.

Дальнейшая работа должна быть связана, прежде всего, с устранением выявленных недостатков и ограничений, связанных с улучшением надежности системы, необходимостью отказа от системы RTK для локализации, улучшением системы планирования и управления для избегания препятствий, а также улучшением качества обнаружения объектов за счет комплексирования данных различных сенсоров.

Список литературы

[Buyval, 2017] Buyval A. et al. Deriving overtaking strategy from nonlinear model predictive control for a race car //2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). – IEEE, 2017. – С. 2623-2628.

[Buyval, 2018] Buyval A. et al. Realtime Vehicle and Pedestrian Tracking for Didi Udacity Self-Driving Car Challenge //2018 IEEE International Conference on Robotics and Automation (ICRA). – IEEE, 2018. – С. 2064-2069.

[DARPA, 2004] DARPA Grand Challenge [Электронный ресурс] // wikipedia.org: Wikipedia, the free encyclopedia, информ.-справочный портал. URL: https://en.wikipedia.org/wiki/DARPA_Grand_Challenge (дата обращения: 01.04.2019).

[Li, 2016] Bo Li. 3d fully convolutional network for vehicle detection in point

[Mousavian, 2017] Mousavian A. et al. 3D bounding box estimation using deep learning and geometry // Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017. 2017. Vol. 2017–Janua. P. 5632–5640.

[OSCC, 2019] DARPA Open Source Car Control [Электронный ресурс] // github.com. URL: https://github.com/PolySync/oscc (дата обращения: 01.04.2019).

[React, 2019] https://reactjs.org/

[Redmon, 2018] Redmon J., Farhadi A. YOLOv3: An Incremental Improvement. 2018.

[Wan et al., 2017] Wan G. et al. Robust and Precise Vehicle Localization based on Multi-sensor Fusion in Diverse City Scenes. 2017.

cloud. arXiv preprint arXiv:1611.08069, 2016.

УДК 62-521

МОДЕЛЬ НАБЛЮДАЕМОСТИ ОБЪЕКТОВ ДЛЯ АВИАЦИОННОГО МОНИТОРИНГА ПОЖАРОВ

Н.В. Ким, М.М. Мокрова (*mary.mokrova@mail.ru*)
МАИ, Москва

Аннотация. В статье предложен подход к выбору оптимальной высоты полета беспилотного летательного аппарата (БЛА) при мониторинге пожарной обстановки на основе использования эвристической модели наблюдаемости искомых объектов. Разработанная модель наблюдаемости основана на учете изменяемой контрастности наблюдаемой сцены от параметров пожара и высоты полета БЛА. Показано, что уменьшение контрастности увеличивает ошибки правильного обнаружения объектов поиска и, соответственно ухудшает эффективность мониторинга.

Ключевые слова: беспилотный летательный аппарат, мониторинг пожаров, наблюдаемость, контрастность, ошибки обнаружения объектов.

Введение

Пожары являются одной из причин самых масштабных по размеру урона катастроф. На текущий момент в современном мире используется множество технологий для предотвращения развития стихийного развития пожаров и их подавления, предотвращения человеческих жертв и сокращения материального ущерба. Для решения данной проблемы эффективным средством является авиационной мониторинг областей, охваченных пожаром, в частности с использованием беспилотных летательных аппаратов (БЛА). Важнейшим показателем эффективности таких операций является достоверность обнаружения объектов поиска (людей, техники и пр.).

В условиях пожара из-за наличия пламени и дыма видимость объектов интереса хуже, чем в нормальных условиях наблюдения. В зависимости от интенсивности дыма, эффективная дальность наблюдения меняется, что приводит к изменениям вероятности правильного обнаружения или распознавания объектов. Требуемой вероятности обнаружения можно добиться, выбирая определенную высоту полета БЛА. Однако управление высотой возможно только при наличии соответствующих моделей

наблюдаемости объектов интереса, формирование которых чрезвычайно сложно.

В данной статье предлагается использовать приближенную эвристическую модель наблюдаемости объектов интереса, учитывающую высоту полета БЛА и некоторые атрибуты пожарной обстановки. Использование описываемой модели может увеличить эффективность поиска и распознавания объектов интереса.

1 Критерий эффективности мониторинга

Эффективность операции обнаружения объектов интереса зависит от вероятности правильного обнаружения $P_d(h, \beta, Y_0|X_0)$, где h – высота полета БЛА, $\beta = (\sigma_s/\sigma_n)^2$ – отношение сигнал/шум, Y_0 – значение принимаемого признака объекта, X_0 – факт присутствия объекта поиска.

Не обнаружение присутствующего объекта (пропуск цели) в условиях пожара является катастрофической ошибкой поиска

$$P_{lo}(h, \beta|X_0) = 1 - P_d(h, \beta, Y_0|X_0). \quad (1)$$

В общем случае минимизация потерь связана со снижением высоты полета БЛА. Однако необходимо учитывать, что влияние дыма и тепловых факторов (на малых высотах полета) могут привести к аварии БЛА.

Примем, что показателем эффективности, на основании которого производится выбор рабочей высоты полета h_r, является уровень допустимой вероятности P_r, связанной с пропуском цели,

$$h_r = \arg [P_{lo}(h, \beta|X_0) \leq P_r]. \quad (2)$$

Таким образом, основной задачей управления становится выбор оптимальной высоты полета БЛА, при которой вероятность обнаружения максимальна или ограничена снизу, а вероятность потери БЛА соответственно ограничена или минимальна.

Для нахождения данной зависимости необходимо описать модель наблюдаемости.

2 Эвристическая модель наблюдаемости объекта интереса

В разработанной модели наблюдаемости учитываются следующие факторы, влияющие на качество получаемого изображения и достоверность обнаружения на нем объектов интереса:

- Условия, в которых производится наблюдение, в том числе характеристики пожара и параметры местности, охваченной огнем.
- Контрастность, обусловленная наличие дыма на наблюдаемой сцене.

Исходя из разработанного вида модели наблюдаемости, можно получить формулу расчета вероятности ошибок обнаружения, а также вид критерия эффективности мониторинга.

2.1 Изменение контрастности изображения

Мониторинг пожарной обстановки происходит в условиях задымленности, что уменьшает контрастность получаемых для обработки, поиска и распознавания изображений. Данный эффект обусловлен тем, что в дыме во взвешенном состоянии находятся мелкие твердые частицы **[Сарманаев и др., 2015]**. Если обследуемая область состоит из F участков, на которых условия пожара различны, то результаты обнаружения объектов будут зависеть от задымленности конкретного участка и высоты полета БЛА.

Для описания зависимости изменения уровня контрастности получаемого изображения от высоты зададимся сигмоидой вида

$$K_f(h)= K_{fmax}[1-1/(1+e^{-k_f (h-h_f)})], \qquad (3)$$

где h – высота полета БЛА, k_f – эмпирический коэффициент, зависящий от условий пожара, атмосферных условий и пр., h_f – высота, при которой контрастность равна 0.5* K_{fmax}, $f \in F$ – индекс области пожара, F – количество участков пожара с постоянными параметрами задымленности, K_{fmax} – максимальная контрастность.

При демонстрации работы предлагаемого алгоритма для большей наглядности будем использовать объект – автомобиль (рисунок 1).

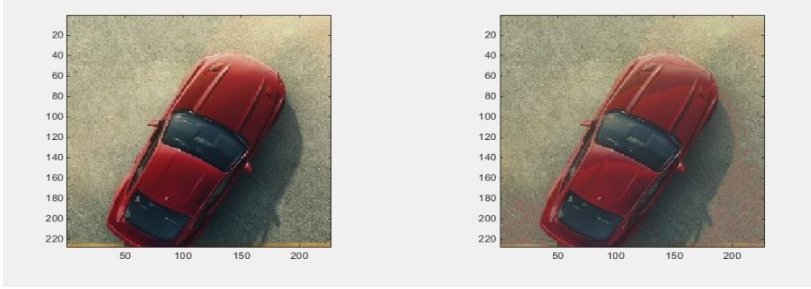


Рис. 1. Понижение контрастности на 0.15.

Однако необходимо заметить, что объект наблюдения может быть любым. В зависимости от вида объекта наблюдения изменяются коэффициенты, выбор которых описан в разделе 2.2 настоящей статьи.

В качестве примера зададимся исходными условиями: k_f = 0.43, K_{fmax} = 3, высота полета h = 4м, h_f = 8м, в таком случае $K_f(4)$=2.54. При данных исходных условиях результат работы модели контрастности изображения показан на рисунке 1. На данном рисунке изображен объект, контрастность которого понижена на 15% от максимальной, т.е. контрастности эталона с текущей местностью на высоте h = 0м.

Таким образом, при мониторинге участка пожара f на высоте h принимаемое изображение сцены

$$B(i, j, h) = B_r[i, j, K_f(h)]+n(i, j), \qquad (4)$$

где $B(i, j, h)$ – яркость в точке изображения с координатами i, j; $B_r[i,j,K_f(h)]$ – математическое ожидание яркости; $n(i, j)$ – белый гауссовский шум с нулевым математическим ожиданием и среднеквадратическим отклонением $\sigma_n(i, j)$.

На рисунке 2 представлены итоговые изображения, используемые для работы модели с уменьшением контрастности изображения, как показано на рисунке 1 данной статьи, и наложением шума при отношении сигнал/шум равном 7 и 5 соответственно.

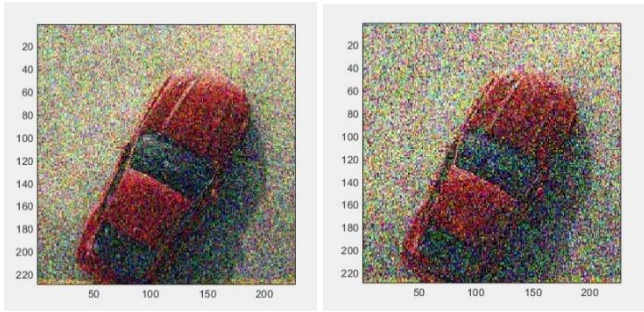


Рис.2. Добавление шума $\beta=7$ и $\beta=5$.

2.2 Условия наблюдения

Значительное влияние на достоверность обнаружения объектов оказывают условия, в которых проводится наблюдение. При различных условиях на одной и той же высоте полета над наблюдаемой сценой вероятность обнаружения различна, что и отражено в модели наблюдаемости в виде коэффициентов условий наблюдения, являющихся параметрами модели наблюдаемости:

- k_f – эмпирический коэффициент, зависящий от условий наблюдения и расположения объектов интереса.
- h_f – высота полета БЛА, при которой достоверность обнаружения людей равна 0.5.

Ниже приведены примеры правил, обеспечивающих оценку параметров моделей наблюдаемости:

- Если поверхность – лес и пожар сильный, и влажность высокая, то $0.7 < k_f < 0.9$.
- Если поверхность – лес и пожар слабый, и лес редкий, и влажность низкая, то $h_f > 0.7\ h_t$, $0.4 < k_f < 0.6$, где h_t – высота деревьев.
- Если поверхность – лес и высота деревьев h_t, то $h_f > h_t$.
- Если поверхность – поле, и пожар сильный, и влажность низкая, то $h_f > 3m$.
- Если поверхность – поле, пожар сильный, и влажность высокая, то $h_f > 5m$.

Коэффициент k_f (размерность 1/метр или $1/m$) и значения высот h_f (размерность - m) могут быть определены на основе сформированных заранее баз знаний исходя из предыдущего опыта обследования аналогичных особых ситуаций.

При отсутствии баз знаний, описываемые выше коэффициенты могут быть выбраны на основе визуального анализа пожарной обстановки. Так на рисунке 3(а) представлена ситуация, когда видимый уровень пламени выше h_t, исходя из этого коэффициент k_f задается в диапазоне $0.7 < k_f < 0.9$, конкретное значение в данном диапазоне выбирается в зависимости от влажности воздуха в данной местности – чем влажность выше, тем коэффициент ближе к значению 0.9.

Уровень наблюдаемости зависит от наличия и интенсивности дыма при пожаре. На представленном изображении дым плотный и его верхняя кромка существенно выше верхней кромки пламени. Параметр h_f выбирается равным 1.2 от высоты пламени.

По изображению местности, представленному на рисунке 3(б), видно, что в данном случае лес редкий, плотность дыма низкая, поэтому коэффициент k_f, принимает значение из диапазона $0.2 < k_f < 0.7$. Параметр h_f определяется исходя из условий $h_f > 0.7\ h_t$.

а)

б)

Рис. 3. (а, б) Примеры пожарной обстановки.

Представленную базу правил для выбора коэффициентов модели наблюдаемости можно расширять, исходя из опыта экспертов и экспериментов, проводимых на натурных моделях.

2.3 Эмпирическая зависимость вероятности

На основании исследований, проведенных методами статистического моделирования, и с учетом (3), (4), была получена следующая эмпирическая зависимость вероятности ошибок обнаружения (1):

$$P_{lo}(h, \beta) = [1-1/(1+e^{-k_f (h-h_f)})]\,[1/(1+e^{-(\beta-\beta_0)})]\,. \quad (5)$$

При различных уровнях β зависимость вероятности ошибок обнаружения от высоты может иметь вид, представленный на рис.4.

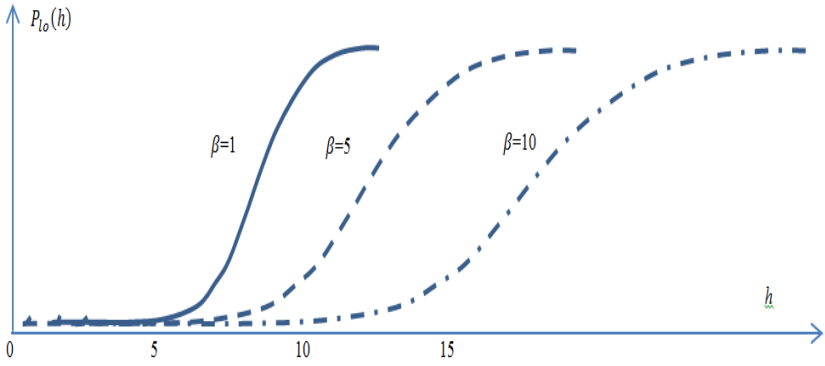


Рис. 4. Вероятности ошибок обнаружения.

Как видно из (5), уровень наблюдаемости зависит не только от β, но также от описанных в разделе 2.2 коэффициентов.

Так при уменьшении коэффициента k_f графики сигмоид, изображенных на рисунке 4, становятся более пологими, что соответствует повышению вероятности правильного обнаружения на одной высоте при улучшении условий наблюдения (увеличению разреженности дыма).

Параметр h_f влияет на сдвиг графика относительно оси P_{lo} влево или вправо при уменьшении или увеличении h_f, соответственно. Следовательно, при рассмотрении полета на одной и той же высоте и при увеличении значения h_f (высоты, обеспечивающей вероятность обнаружения равную 0,5) вероятность ошибки уменьшается.

Представленная в формуле (5) зависимость может быть использована для расчета рабочей высоты полета h_r (2) авиационного мониторинга пожара, описываемого в разделе 1 настоящей статьи.

Так, для принятых ранее параметров пожара и допустимой вероятности ошибки $P_r = 0{,}1$ и $\beta = 5$, высота полета должна быть $h_r \le 6$ м, а для $\beta = 10$, $h_r \le 12$ м, что видно из представленного выше рисунка 3.

Таким образом, проведение мониторинга на высоте $h_r = 12$ м при $\beta = 10$ обеспечивает $P_{lo}(h, \beta|X_0) = 0{,}1$, но если при $\beta = 5$ мониторинг проводится на высоте $h_r = 12$ м, то вероятность пропуска цели составит $P_{lo}(h, \beta|X_0) = 0{,}5$,

что показывает существенное ухудшение эффективности поиска объектов интереса.

Заключение

В работе предложена эвристическая модель наблюдаемости объектов интереса, учитывающая высоту полета БЛА и некоторые атрибуты пожарной обстановки. Использование описываемой модели позволяет увеличить эффективность поиска и распознавания объектов интереса за счет выбора необходимой высоты полета, в соответствии с условием (2).

Предложенная эвристическая модель может быть использована при выборе рабочей высоты полета БЛА при проведении мониторинга пожарной обстановки. Моделирование показало, что данная модель наблюдаемости устойчива и эффективна при осуществлении поиска и распознавания объектов интереса.

Список литературы

[Гостев, 2010] Гостев И.М. О влиянии точности представления изображений на качество распознавания графических образов // Вестник РУДН. Серия: Математика, информатика, физика. 2010. №3-2. – https://cyberleninka.ru/article/n/o-vliyanii-tochnosti-predstavleniya-izobrazheniy-na-kachestvo-raspoznavaniya-graficheskih-obrazov (дата обращения: 04.02.2019).

[Сарманаев и др., 2015] Сарманаев С.Х., Башарин В.А., Толкач П.Г., Шербашов К.А. Токсико-химическое поражение на пожаре // Биомедицинский журнал «Medline.ru», том 16, токсикология, 26 марта 2015. – http://www.medline.ru/public/pdf/16_041.pdf

[Kim et al, 2018] Kim N., Bodunkov N. «Computer Vision in Control Systems - 3: Aerial and Satellite Image Processing», Volume 3, Editors M. Favorskaya, Lakhmi C. Jain, Springer 2018. – 343 p.

[Евдокименков и др., 2018] Евдокименков В.Н., Ким Н.В., Мокрова М.И. Мониторинг пожарной обстановки беспилотными летательными аппаратами. Материалы XI Всероссийской студенческой научно-технической школы-семинара «Аэрокосмическая декада». – М., 2018. – С.30

[Мокрова, 2018] Мокрова М.И. Алгоритм оптимального выбора высоты полета и разделения функциональной группы БЛА при мониторинге пожарной обстановки. Тезисы докладов IV Общероссийской молодежной научно-технической конференции. СПб., 2018. – С.57.

УДК 519.878, 004.896

ПОИСК НАЗЕМНЫХ ОБЪЕКТОВ ГРУППОЙ БЕСПИЛОТНЫХ ЛЕТАТЕЛЬНЫХ АППАРАТОВ С ИПОЛЬЗОВАНИЕМ ЭНТРОПИЙНОГО ПОДХОДА

Н.А. Михайлов (*n.mikahylov.mai@mail.ru*)
Московский авиационный институт (национальный исследовательский университет), Москва

Аннотация. В статье представлен алгоритм согласованного поиска группой беспилотных летательных аппаратов. В основе алгоритма лежит вычисления энтропийной карты местности для множества целей. Критерием выбора траектории следования является максимум суммарной пропускной способности. В работе представлены результаты компьютерного моделирования предложенного алгоритма. Показано повышение производительности решения задачи в сравнении с решением аналогичной задачи алгоритмом поиска по максимальной априорной вероятности присутствия объектов.[1]

Ключевые слова: поиск объектов, группа БЛА, информационная энтропия, планирование маршрута.

Введение

Задача поиска наземных объектов является актуальной на сегодняшний день. Особый интерес представляет решение задачи вторичного поиска. Основной особенностью вторичного поиска является наличие известной априорной вероятности возможного присутствия объекта поиска в некоторой области. Эта особенность позволяет направлять поисковые силы в области максимальной вероятности, что повышает эффективность поиска. Примерами задачи вторичного поиска являются:

- Поиск пострадавших в районах стихийного бедствия;
- Поиск очагов лесных пожаров;
- и др.

От эффективности и надежности решения задачи вторичного поиска, в приведенных выше примерах, могут зависеть человеческие жизни.

[1] Работа выполнена при поддержке РФФИ (проект № 19-08-00613-а)

В работах [Амелин и др. 2009], [Kim et al. 2015] представлены варианты решения задачи поиска наземных объектов с использованием беспилотных летательных аппаратов (БЛА).

В работе [Михайлов, 2017] было показано, что учет фактора наблюдаемости в области поиска, таких как: туман, дым, загораживание обломками или деревьями, - повышает эффективность решения задачи поиска наземных объектов.

Однако в известных работах не рассматривалась задача поиска множества наземных объектов группой БЛА с учетом фактора наблюдаемости.

1 Постановка задачи

Проблема планирования поиска группой БЛА состоит в том, что при организации вторичного поиска полет строем в ряде случаев является избыточным. Целесообразно распределять поисковые ресурсы в различные области пространства поиска.

Таким образом, целью работы является повышение эффективности поиска наземных объектов за счет разработки методики планирования маршрутов движения БЛА в составе группы, при которой учитывались бы факторы: априорная вероятность присутствия объекта поиска в заданной области, наблюдаемость объекта поиска в заданной области, а также взаимное расположение БЛА и объекта поиска.

Как было показано в работе [Ким и др., 2018] необходимо задать априорную вероятность присутствия объекта поиска в области поиска с размерами X_{MAX}, Y_{MAX}. В случае множества объектов поиска M данные вероятности задаются в виде $P_m(x, y)$ где $m \in M$; x, y – координаты.

Для обнаружения объекта необходимо задать набор признаков $\bar{u} = \{u_1 \in U_1, \ldots, u_R \in U_R\}$, где R общее количество признаков, к которым можно отнести: яркость, цвет, текстуру, размеры, форму и др. Наиболее полным описанием признаков является условная плотность распределения вероятности возникновения значения признака u_r, при условии нахождения или отстутствия объекта поиска [Ким, 2001] .

В работе [Бодунков, 2015] показано, что данные плотности могут быть построены с учетом различных факторов: погодные условия, время суток, время года и др.

В качестве критерия эффективности решения задачи поиска в работе приятно минимальное время поиска $T_{\text{П}}$ при заданной минимальной вероятности правильного обнаружения:

$$\min_t T \text{ при } P_{\text{по}} \text{ і } P_{\text{тр}} \quad \textbf{(2)}$$

Требуется разработать методику планирования траекторий для каждого БЛА в составе группы, обеспечивающей минимум заданного критерия.

2 Алгоритм согласованного поиска группой

Пусть, на каждом шаге процесса формирования поисковой траектории каждый БЛА обладает полной информации о положении других БЛА, текущем состоянии наблюдаемой сцены и вероятностях присутствия объектов поиска в области. Зададим ограничения, что в каждой точке области поиска одновременно может присутствовать лишь один БЛА. Тогда, в момент начала операции каждый БЛА рассчитывает критерий пропускной способности [Коган, 1981] $C_m(x, y)$ для каждого объекта поиска *m*. Далее необходимо выбрать оптимальное распределение БЛА по зонам максимумов пропускной способности. С учетом введённого выше ограничения, необходимо выбрать такое распределение, при котором доставлялся бы максимум суммарной пропускной способности. Таким образом, если количество БЛА обозначить как N, то количество вариантов распределения будет равно $q = N \cdot N$, а оптимальный план q^* будем находить из условия:

$$C_\Sigma\left(q^*\right) = \max_q \sum_{n=1}^{N} C_n(q), \quad \textbf{(3)}$$

где $C_n(q) = \max_{x,y} C_m(x, y)$ – максимальное значение пропускной способности для БЛА под номером *n*. Для учета ограничения необходимо при составлении плана для БЛА под номером *n* исключать точки *x, y*, которые были выбраны аппаратами с индексами [1,*n*-1].

После составления плана осуществляется движение в сторону выбранных точек. При достижении одним из БЛА плановой точки, происходит решение задачи обнаружения. В результате решения задачи обнаружения в соответствии с [Михайлов, 2017] изменяется распределение вероятностей присутствия объекта поиска за счет получения нового значения $P_m(x, y) = P_{\text{ПО}}(x, y)$. Если $P_{\text{по}}$ і $P_{\text{тр}}$ то объект считается обнаруженным. Если объект не удалось обнаружить, или обнаруженный объект был не последним, то данная информация попадает на борт остальных участников поиска. Далее происходит вторичное определение плана с учетом текущего положения БЛА и новых значений $P_m(x, y)$. Вся процедура распределения, полета и обнаружения продолжается до тех пор,

пока не будет обнаружены все объекты поиска или не останется областей, для успешного решения задачи обнаружения [Ким, 2001].

3 Компьютерное моделирование

Для проверки работоспособности алгоритма и оценки его эффективности было проведено компьютерное моделирование. Группа БЛА состояла из $N = 4$ агентов. Область поиска составляет X_{MAX} г $Y_{MAX} = 50$г 50. В области поиска необходимо обнаружить $M = 10$ объектов. Для каждого объекта имеется распределение $P_m(x, y)$, причем суммарное значение вероятностей на всей карте поиска равно единице.

Было проведено две серии экспериментов: «хорошая» наблюдаемость и «плохая» наблюдаемость. «Хорошая» наблюдаемость состоит в том, что для каждой точки карты выполняется условие:

$$P(O(x, y) \mid \bar{u}^*) \geq 0.9, \qquad \textbf{(5)}$$

где $P(O(x, y) \mid \bar{u}^*)$ – вероятность правильного обнаружения при получении значения признака $\bar{u}^*$ в точке с координатами *x, y* ; $\bar{u}^*$ – значение признака соответствующее области правильного обнаружения [Ким, 2001]. Другими словами, при нахождении объекта поиска в точке *x, y* он будет обнаружен с вероятностью $P \geq 0.9$.

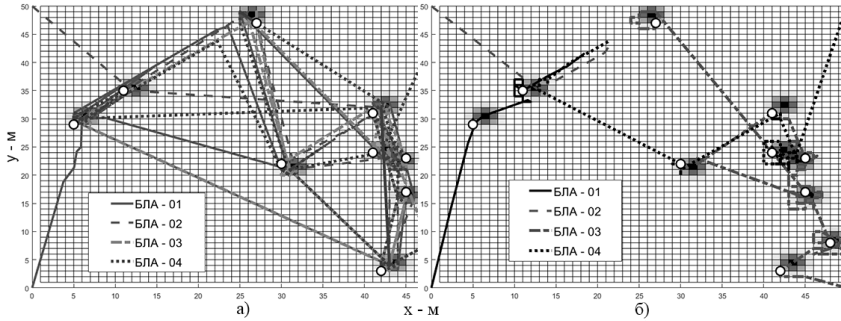


Рис. 1. Траектории поиска группой БЛА: а) траектории построенный с использованием алгоритма поиска по максимальной вероятности; б) траектории построенный с использованием энтропийного подхода.

Для эксперимента с «плохой» наблюдаемостью в некоторой точке, в которой находится случайно выбранный объект, выполняется условие:

$$P(O(x, y) \mid \bar{u}^*) < 0.55 \qquad \textbf{(6)}$$

Начальные положения БЛА: БЛА1 (0,0); БЛА2 (50,0); БЛА3 (0,50); БЛА4 (50,50). На рисунке (рис. 1) показаны конечные траектории поиска для алгоритма поиска по максимальной вероятности присутствия (рис. 1, а) и энтропийного поиска (рис. 1, б) при условии «хорошей» наблюдаемости. Кружками отмечены фактические положения объектов поиска на фоне распределения вероятностей присутствия (чем темнее, тем выше вероятность).

На рисунке (рис 2.) показана оценка эффективности поиска, которая проводилась на основе изменения энтропии положения объектов поиска от суммарного затраченного времени поиска.

По результатам моделирования поиска при «хорошей» наблюдаемости видно существенное различие в длине траекторий, это объясняется тем, что при выборе следующей точки пути энтропийный алгоритм учитывает расстояние до ближайших областей поиска. Из графиков (рис 2.) для случая «хорошей» наблюдаемости суммарное время поиска составило 119.9 секунд, в то время как поиск по максимальной вероятности потребовал 178.1 секунду. Превышение затрат времени на 48%.

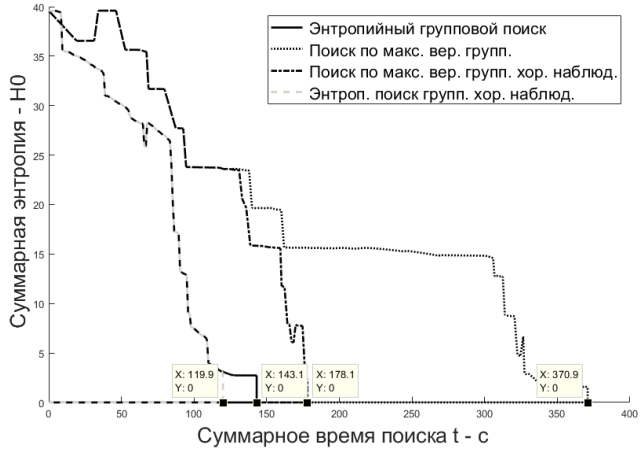


Рис. 2. Изменение энтропии при поиске группой БЛА.

На графиках (рис 2.) также представлены результаты при «плохой» наблюдаемости. Время энтропийного поиска составляет 143 секунды, а время поиска по максимальной вероятности составило 370.9 секунды. Превышение затрат времени на 159%. Таким образом, энтропийный подход обеспечивает существенное снижение времени группового поиска по сравнению с методом планирования по максимальной вероятности.

4 Выводы

Предложена методика планирования траекторий для поиска наземных объектов группой БЛА. Основные особенности алгоритма: учет фактора наблюдаемости объектов поиска, учет вероятности присутствия каждого объекта поиска, а также учет расстояния между текущим положением БЛА и предполагаемыми точками поиска (расстояние может быть заменено на некоторую функцию потерь, учитывающую и другие важные факторы при поиске).

Результаты моделирования подтверждают работоспособность и эффективность предлагаемой методики при групповом применении БЛА. Тестовые примеры показали повышение эффективности поиска, в случаях «плохой» видимости, в более чем 1.5 раза.

Список литературы

[Амелин и др. 2009] Амелин К.С., Антал Е.И., Васильев В.И., Граничина Н.О. Адаптивное управление автономной группой беспилотных летательных аппаратов // Стохастическая оптимизация в информатике; Изд-во СПБГУ, т. 5, №1, 2009, стр. 157-166.

[Kim N, 2015] Kim N., Chervonenkis M. Situational control unmanned aerial vehicles for traffic monitoring. // Modern Applied Science, 2015, Vol. 9, No. 5, Special Issue//Canadian Center of Science and Education. ISSN (printed): 1913-1844. ISSN (electronic): 1913-1852

[Ким и др., 2018] Ким Н.В., Михайлов Н.А. Энтропийный подход к решению задачи поиска наземных объектов с учетом фактора наблюдаемости объекта поиска// Техническое зрение в системах управления – 2018. Сборник тезисов; Москва, 2018, стр. 59.

[Михайлов, 2017] Михайлов Н.А. Планирование маршрута поиска для автономного беспилотного летательного аппарата с использованием энтропийного подхода // сборник трудов семинара, IV Всероссийский научно-практический семинар Беспилотные транспортные средства с элементами искусственного интеллекта, 5-6 октября 2017, Казань, Казанский (Приволжский) университет, стр. 126-135.

[Коган, 1981] Коган И.М., Прикладная теория информации // М.: Радио и связь. 198., 216 с.

[Бодунков, 2015]. Бодунков Н.Е. Расширение условий функционирования систем визуальной навигации автономных беспилотных летательных аппаратов // Диссертация на соискание ученой степени кандидата технических наук. Московский авиационный институт (технический университет), 2015, 155 стр.

[Ким, 2001] Ким Н.В. Обработка и анализ изображений в системах технического зрения: Учебное пособие. // М.: Изд-во МАИ, 2001 – 164 с

УДК 528.71

РАЗРАБОТКА БПЛА МУЛЬТИКОПТЕРНОГО ТИПА ДЛЯ ПОИСКА ЛЮДЕЙ В ЛЕСНЫХ МАССИВАХ

П.М. Трефилов (petertrfi@gmail.com)
Институт Проблем Управления, Москва

Р.В. Мещеряков
Институт Проблем Управления, Москва

А.В. Чехов
Институт Проблем Управления, Москва

Аннотация. В статье рассмотрена возможность применения БПЛА для поиска человека в лесных массивах. Приведен анализ существующих решений, предложена своя концепция. Описаны преимущества предложенной концепции и алгоритм действий.

Ключевые слова: поиск человека, БПЛА, распознавание людей, мультикоптер.

Введение

По данным МВД, в России ежегодно в розыске находятся свыше 120 тысяч без вести пропавших [Владыкина, 2012]. Настолько большая цифра обусловлена не только похищением людей, но и заблудившимися людьми в лесу. Технология поиска пропавшего человека – это методичное прочесывание леса с одновременным созданием громких звуков для привлечения внимания. Вместе с тем, прочесывание леса поисковым отрядом, составленным, как правило, из добровольцев – длительный и ресурсоемкий процесс, что ограничивает его применение, особенно в отдалённых районах, на больших площадях и в сложных погодных условиях. В подавляющем большинстве случаев, люди, которые прочесывают лес – это добровольцы или волонтеры. К сожалению, на сегодняшний день – это единственный надежный способ для поиска человека. Иногда службы спасения применяют вертолеты, но в условиях лесных массивов довольно сложно разглядеть человека под кромкой деревьев. Для того, чтобы человек был замечен вертолетом – ему необходимо выйти на равнинную местность, не закрытую деревьями и подать сигнал. Из-за того, что подъем вертолета не гарантирует успех в

поисковой операции и ресурсозатратен – его применяют в исключительных случаях.

К сожалению, поиск людей – это операция, которая требует очень большого количества людей, терпения, сноровки, а также достаточной материальной основы: вода, еда, спецодежда для различных погодных условий и др.

1 Применение беспилотных летательных аппаратов для поисковых работ

1.1 Постановка задачи

Для вывода технологии поиска людей на новый уровень, с минимизацией ресурсов и привлечения добровольцев компания «АФК СИСТЕМА» проводит конкурс Одиссея [Томчук 2018]. Задачей конкурса, в рамках которого выполняется проект – является создание устройства или технологии, способных обнаружить точные координаты местонахождения пропавшего человека в радиусе не менее 10 км в природной среде (лес) в течение не более 10 часов при отсутствии источников связи, вне зависимости от времени суток и погодных условий.

1.2 Подход к решению

Оптимальным решением поставленной задачи может стать применение беспилотных летательных аппаратов (БПЛА). Применение БПЛА значительно дешевле, чем взлет вертолета или использование самолета. Управление БПЛА не требует большого количества людей и возможна автоматизация процесса поиска человека. Площадь обследуемых площадей значительно превышает ту, которую сможет пройти поисковая группа людей.

На сегодняшний день разделяют два вида управления БПЛА – ручное и автоматическое.

При использовании БПЛА в ручном режиме возможно точное обследование необходимой местности. Такой метод не исключает человеческого фактора, например, неправильная команда способна привести к поломке. Поэтому главной сложностью при применении подобного БПЛА – наличие квалифицированного оператора, способного управлять объектом.

В автоматическом режиме БПЛА задается команда, которую он выполняет без участия оператора, например, облет леса. Для использования БПЛА с автоматическим управлением необходимо задать обследуемую область, а также предполагаемый маршрут. Таким образом возможно исключение необследованных зон.

Для повышения точности в поисковых операциях с применением БПЛА возможно применение технических устройств таких, как тепловые визиры или Инфракрасные (ИК) визиры. Их применение позволит обнаружить человека издалека, а применимые визиры способны работать как днем, так и ночью.

Ключевым недостатком при использовании визиров является отражение и обратное рассеяние от среды (запыленность воздуха, дождевые капли, снег, туман), особенно в ближней зоне, когда размеры снежинок капель сравнимы с апертурой объектива. Это ограничивает их применение при облете леса, особенно в жаркое время года. Листья деревьев, нагретые солнечным светом, не позволят сенсорам визировать земную поверхность.

Для того, чтобы исключить этот недостаток необходимо применение БПЛА под кромкой лесного массива. Такое применение накладывает ряд ограничений на БПЛА.

В первую очередь, это ограничения по габаритным характеристикам. Аппарат не должен быть громоздким и обладать достаточной маневренностью для предотвращения столкновений. В данном случае оптимальным решением будет использование БПЛА мультикоптерного типа.

Значительно усложняется управление БПЛА в ручном режиме. Сложность при маневрировании ограничивает скорость полета летательного аппарата, что сказывается на размере обследуемой территории, а незамеченная оператором ветка дерева может стать фатальной для БПЛА.

При применении БПЛА с автоматической системой управления необходимо разработать такой комплекс пилотажно-навигационного оборудования, который бы позволял заметить препятствия и принять меры для предотвращения столкновения.

2 Принцип работы БПЛА

Для реализации поставленной задачи командой «GoFPV 2019» ведется разработка прототипа модели, в которой применяется рой БПЛА мультикоптерного типа с автоматической системой предупреждения столкновений. Такой подход позволяет увеличить скорость прочесывания леса, а также уменьшит риски поисковой миссии при выходе из строя одного из аппаратов.

2.1 Методика

Для реализации описанного сценария требуется, по предварительной оценке, 8-10 БПЛА мультикоптерного типа и 2 оператора с ноутбуками, находящиеся на противоположных сторонах прочесываемого квадрата.

Задачи операторов – запустить БПЛА, заменить аккумулятор и карту памяти прилетевшего, запустить новый БПЛА в случае потери одного из роя, запустить обработку отснятых материалов на ноутбуке с нейросетью, сигнализировать об обнаружении человека по результатам обработки видео. Описанный метод изображен на рисунке 1.

В качестве дополнительной меры повышения надежности и сокращения времени поиска прорабатываются варианты трансляции видеопотока на рабочую станцию для распознавания нейросетью и мониторинга оператором.

БПЛА летят автономно по GPS в соответствии с составленным для них полетным заданием. Маршрут и высота могут варьировать, в зависимости от характера местности (возможно применение эвристик, сокращающих время полета).

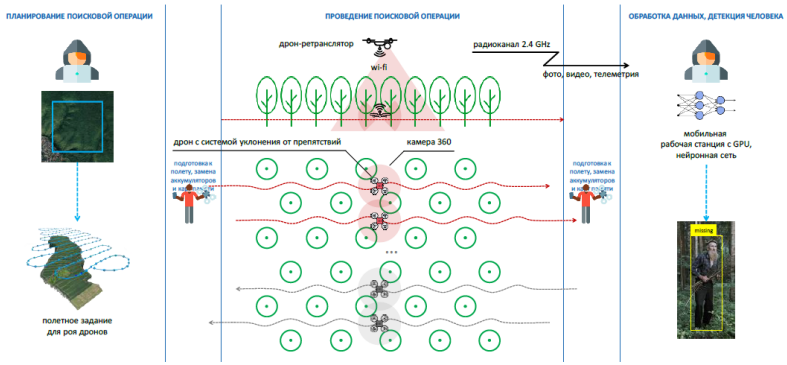


Рис. 1. Схема работы БПЛА команды «GoFPV 2019».

2.2 Система предупреждения столкновения от препятствий БПЛА

Как было описано ранее, БПЛА должен обладать автоматической системой управления. Для этого мультикоптер оборудован полетным контроллером Pixhawk 4. Оператор формирует полетное задание – это прямая траектория с набором ключевых точек, расположенных в 100 м друг от друга. БПЛА движется по траектории от точки к точке, тем самым выполняя задание. Расположенные точки имеют допустимую область 3 метра, а также не имеют привязки по высоте, так как невозможно учесть рельеф местности.

БПЛА совершает полет под кромкой леса, поэтому необходимо применять такую систему предотвращения столкновений, которая должна

быть быстродейственной, точной и надежной. Алгоритм уклонения от препятствий должен заблаговременно обнаруживать даже мелкие ветки, а уклонение должно быть оперативным.

Для реализации системы предотвращения столкновений применяется стереокамера Intel Realsense D435 [Carrio et al., 2018]. С помощью камеры формируется карта глубин местности. Затвор камеры имеет угол обзора 85,2°x58° и дальность до 10м. Камера расположена на фронтальной части БПЛА, а угол ее обзора позволяет получать достаточную карту глубины для прямолинейного движения. Измерения должны проводиться постоянно при перемещении БПЛА, а вычислитель должен учитывать вибрации, которые возникают как при движении мультикоптера, так и при неточности измерений стереокамер.

Для уклонения от препятствий применяется метод потенциальных полей [Медведв и др., 2017]. Его суть состоит в следующем: мультикоптер принимается за материальную точку в пространстве с нулевым потенциалом. Следующая ключевая точка в полетном задании имеет притягивающее потенциальное поле, а каждое препятствие создает отталкивающее потенциальное поле, сила которого обратно пропорциональна расстоянию до него. Таким образом, БПЛА двигается в результирующем направлении.

Алгоритм работы системы предупреждения столкновений представлено на рисунке 2.

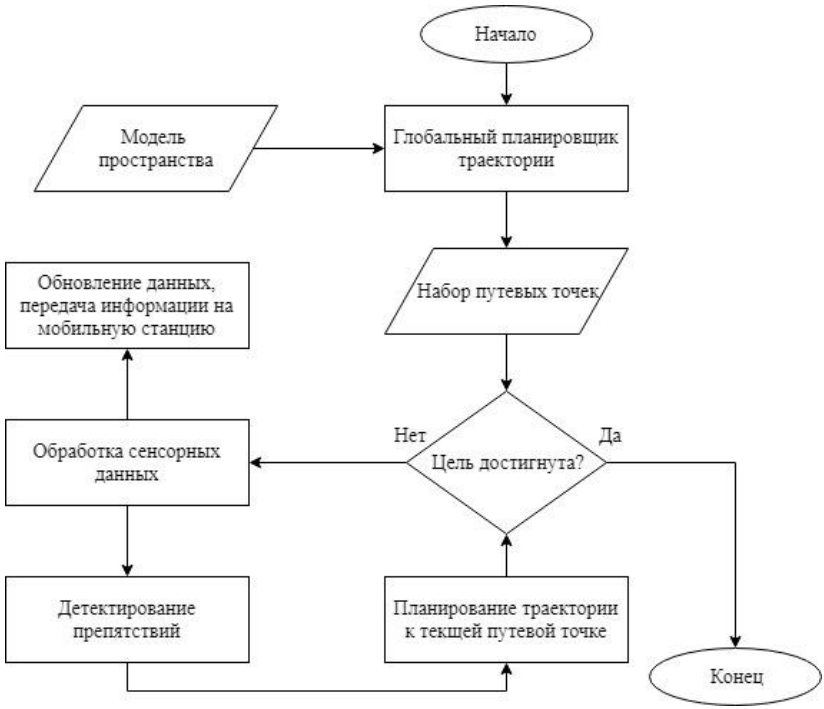


Рис. 2. Алгоритм работы системы предотвращения от препятствий

2.3 Групповое управление

В полетном контроллере головного устройства заданы метки, по которым должен передвигаться мультикоптер. На данном этапе ведутся тестирования с помощью имитационного моделирования [Галин, 2018]. Через порты MAVLink передается информация о местоположении GPS от «главного» БПЛА к остальным. Главным БПЛА выступает центральный мультикоптер в цепи. Положение GPS «главного» БПЛА корректируется с помощью установленного смещения, а затем отправляется остальным в виде серии динамических путевых точек. Таким образом, все БПЛА будут следовать за «главным» на заданных расстояниях смещения.

При выходе из строя «главного» БПЛА его роль занимает ближайший ЛА.

3 Алгоритм распознавания людей

В качестве поиска людей применяется подвешенная к БПЛА камера на стабилизированной платформе. С помощью нейронных сетей камера

распознает образы человека [Dwivedi 2019]. Обучение нейронной сети происходит с применением размеченного датасета [Joseph et al., 2018]. Из него выделяются кластеры для распознавания человека, чтобы не перегружать вычислитель другими элементами. Обучение сверточной нейронной сети происходит с помощью алгоритма обратного распространения ошибки. Примеры распознавания представлены на рисунке 3.

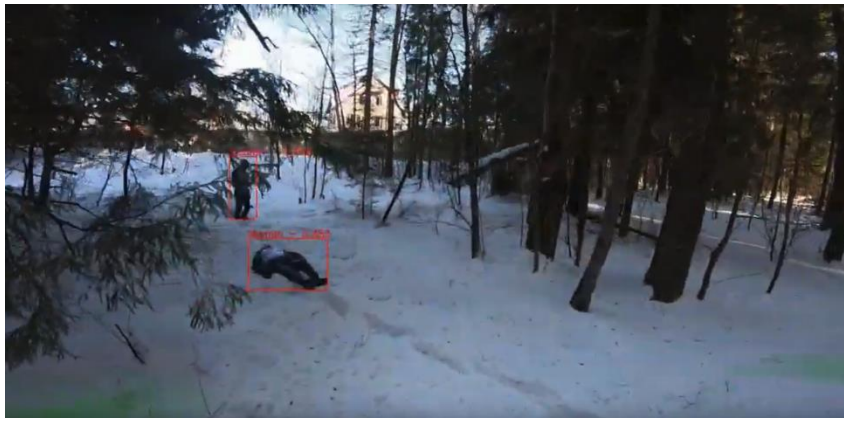


Рис. 3. Примеры распознавания людей

При обнаружении человека БПЛА сигнализирует оператору о найденном совпадении. Человек верифицирует полученные данные, после чего принимается решение о продолжении или окончании поисковой операции.

4 Заключение

Поиск человека трудозатрадная и энергоемкая задача. Значимость работы кране высока, а решаемая задача является в нашей стране острой и актуальной. Предложенный метод решения имеет значительное преимущество по сравнению с вертолетом или цепочкой людей. Командой «GoFPV 2019» уже реализован подход к определению человека, ведутся тестирования системы предотвращения столкновений. В дальнейшем планируется выполнение обработки видео на борту, повышения автономности БПЛА для того, чтобы не было необходимости во втором операторе, а также возможность использования привязной платформы для обеспечения связи и формирования локальной системы навигации.

Благодарности. Авторы считают своим приятным долгом поблагодарить участников Школы Дронов Московского Авиационного Института.

Список литературы

[Владыкина, 2012] Владыкина Т. Российская газета - Федеральный выпуск № 221(5894). – https://rg.ru/2012/09/26/poisk.html

[Галин, 2018] Галин Р.Р. Виртуальный полигон для эффективного взаимодействия роботов в многоагентной робототехнической системе // Известия Кабардино-Балкарского научного центра РАН. – № 6 (86), Часть II. – 2018. С. 112-118.

[Исхаков и др. 2018] Исхаков А.Ю., Диане С.А., Исхакова А.О. Разработка мультиагентной технологии экологического картографирования мегаполиса // Труды 2-й Международной научной конференции "Модели мышления и интеграция информационно-управляющих систем" (ММИИУС - 2018, Нальчик-Терскол). Нальчик - Терскол: Редакционно-издательский отдел КБНЦ РАН, 2018. С. 145-151.

[Медведев, 2018] М.Ю. Медведев, В.С. Лазарев Метод планирования движения группы подвижных объектов с использованием динамических репеллеров и целераспределения // Научный вестник НГТУтом 66, № 1, 2017, с. 41–52

[Томчук, 2018] Томчук Д. – Руководитель конкурса Одиссея - https://www.odyssey.community/

[Brett et al., 2017] Brett T. Lopez, Jonathan P. Aggressive 3-D Collision Avoidance for High-Speed Navigation // IEEE International Conference on Robotics and Automation (ICRA) Singapore, May 29 - June 3, 2017

[Carrio et al., 2018] Adrian Carrio, Sai Vemprala, Andres Ripoll, Srikanth Saripalli, Drone Detection Using Depth Maps // Pascual Campoy IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2018), Madrid, Spain

[Dwivedi, 2019] Priya Dwivedi Pedestrian detection in Aerial Images using RetinaNet - https://towardsdatascience.com/@priya.dwivedi

[Joseph et al., 2018] Joseph Redmon, Ali Farhadi YOLOv3: An Incremental Improvement // University of Washington

УДК 001.57, 004.852, 004.896, 004.942, 519.876.5

ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ И ИНТЕЛЛЕКТУАЛЬНАЯ ОПТИМИЗАЦИЯ ЛОКАЛИЗАЦИИ СТАНЦИЙ ОБСЛУЖИВАНИЯ БПЛА

И.Ю. Данилов (*danilov.iiu@kgeu.ru*)
Казанский государственный энергетический университет,
Казань

И.М. Афанасьев (*i.afanasyev@innopolis.ru*)
Университет Иннополис, Иннополис

Аннотация. Увеличение длительности полёта беспилотных летательных аппаратов (БПЛА) является важной задачей мобильной робототехники. Для достижения этой цели может использоваться автоматизация перезарядки БПЛА с помощью наземных станций обслуживания. Подобный робототехнический комплекс (станция и БПЛА) востребован для охраны и мониторинга объектов различной инфраструктуры. Однако, эффективность системы зависит от оптимального количества станций и их расположения для мониторинга заданной площади. Статья предлагает интеллектуальный алгоритм расчета искомых параметров, основанный на комбинировании имитационного моделирования и генетического алгоритма поиска. Входами алгоритма являются координаты зоны мониторинга и областей, запретных для полетов БПЛА, вероятности проникновения нарушителей на охраняемую территорию через внешний периметр, информация о скорости движения и зарядки БПЛА, количестве дронов, хранимых и обслуживаемых на одной станции, скорость движения нарушителя и частота инцидентов. В ходе многократного имитационного моделирования процесса вторжения в рассматриваемую область и перехвата цели с помощью БПЛА, происходит оптимизация расположения станций обслуживания. Данный подход следует парадигме обучения с подкреплением. В качестве функции оптимизации используется соотношение эффективного полетного времени для сопровождения цели к общему времени, проведенному всеми БПЛА в воздухе. На основе заданного значения критерия оптимизации происходит подбор минимального числа станций, способных решать требуемую задачу охраны и мониторинга.

Ключевые слова: беспилотный летающий аппарат (БПЛА), наземная станция обслуживания, имитационное моделирование, генетический алгоритм поиска, обучение с подкреплением

Введение

В настоящее время широкое использование летательных аппаратов вертикального взлета и посадки (Vertical Take-Off and Landing, VTOL) ограничивается недостаточным временем их пребывания в воздухе [Norouzi Ghazbi et al., 2016]. Одним из подходов при решении этой проблемы является создание роботизированного комплекса, состоящего из летательного аппарата и наземной станции обслуживания, автоматически осуществляющей зарядку/замену элемента питания дрона (Рис. 1).

Рис. 1 Автоматизированная наземная станция обслуживания БПЛА вертикального взлета и посадки

Имитационное моделирование мобильных объектов, хотя и имеет некоторые ограничения [Умников и др., 2018], но часто используется в исследованиях анализа движения в 3D симуляторах, таких как Gazebo [Afanasyev et al., 2015; Shimchik et al., 2016; Sokolov et al., 2017], MATLAB/Simulink [Khusainov et al., 2016], среды ROS/RViz [Ibragimov et al., 2017; Bokovoy et al., 2018, Filipenko et al., 2018], гоночных симуляторов [Zubov et al, 2018] и прикладных программ для захвата движения и генерации соответствующих моделей [Gabbasov et al., 2015]. Различные задачи мониторинга на базе БПЛА исследуются в работах [Афанасьев и др., 2015; Vu et al., 2018; Sabirova et al., 2019], и решения по автоматизированным системам обслуживания БПЛА путем зарядки/замены батарей анализируются в исследованиях [Данилов и др., 2016; Нго и др., 2017]. Применение подобных комплексов имеет большие перспективы, поскольку позволяет автоматизировать использование БПЛА в целях мониторинга и охраны объектов различной инфраструктуры, обработки посевов и сельхозугодий, доставки малогабаритных грузов, мониторинга дорожной обстановки и т.д. [Нго и др., 2017]. Тем не менее,

остаются открытыми вопросы эффективного применения таких систем, в частности, оптимального расположения станций и минимально необходимого их количества для организации мониторинга заданной площади. В работе описывается подход на базе математического моделирования, который позволяет ответить на эти вопросы.

1. Роботизированная станция обслуживания БПЛА

Наземная станция представляет собой роботизированный комплекс, способный хранить БПЛА внутри себя продолжительное время, поддерживая заданную температуру и влажность, за счет наличия систем обогрева, охлаждения и осушки воздуха. При этом внешние погодные условия могут быть жесткими: камера хранения проектируется так, чтобы температура снаружи могла меняться от -40 до +50 С, а влажность достигать 100%, что отличает проектируемую станцию для российских условий от аналогичной разработки компании Airobotics [Airobotics, 2018]. Если во время выполнения задания у дрона заканчивается заряд батареи, он автоматически возвращается на станцию, которая принимает его, и осуществляет обслуживание. После посадки БПЛА, внутри ангара восстанавливаются заданные значения температуры и влажности. Всё это происходит в автоматическом режиме без участия человека.

2. Описание подхода

Для решения задачи оптимального размещения станций в зоне мониторинга используется подход, который совмещает имитационное моделирование и оптимизационный алгоритм (Рис.2). Данный подход следует парадигме «обучения с подкреплением» (Reinforcement learning). В качестве среды моделирования выступает имитационный симулятор, а в качестве оптимизационного алгоритма выбран генетический алгоритм поиска. В результате работы программы происходит итеративный поиск оптимальных значений. В начале, искомые параметры, формирующие так называемую «хромосому», выбираются случайным образом. Прогон каждой из хромосом через имитационный симулятор позволяет получить обратную связь в виде значения целевой функции и оценить качество модели. Генетическая селекция, отбор лучших хромосом и их модификация, направленная на улучшение функции приспособления, позволяет последовательно, шаг за шагом, достигнуть оптимальных значений искомых параметров.

2.1 Процесс имитационного моделирования эксперимента

В ходе многократного имитационного моделирования процесса вторжения в рассматриваемую область и перехвата цели с помощью БПЛА, происходит оптимизация расположения станций обслуживания. В качестве функции оптимизации F_i используется соотношение эффективного полетного времени (таким считается время, потраченное на сопровождение цели) к общему времени, проведенному всеми БПЛА в воздухе.

$$F_i = \frac{\sum_{\text{по всем вылетам всех ЛА}} t_j^{\text{эфф}}}{\sum_{\text{по всем вылетам всех ЛА}} t_j^{\text{общ}}}$$

При решении задачи мы исходим из предположения, что целью мониторинга является определение типа объекта, нарушившего границу, и его сопровождение. Для этого БПЛА использует бортовую камеру, сигнал которой передается на пульт управления мониторинга и охраны. Задача считается выполненной, если нарушитель попал в поле зрения камеры. С точки зрения симуляции это означает, что нарушитель оказывается в границах квадрата, центр которого совпадает с положением дрона. Длина стороны квадрата является настраиваемым параметром. При этом БПЛА движется так, чтобы совместить свою координату с координатой объекта-нарушителя.

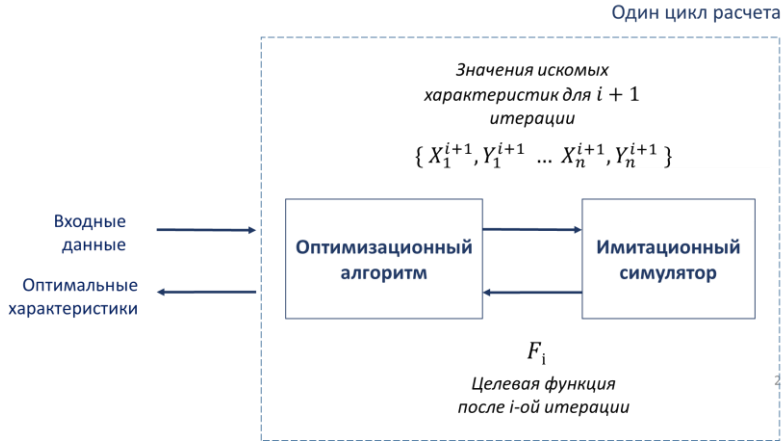


Рис. 2. Блок-схема предложенного подхода к решению задачи оптимального размещения станций, совмещающего имитационное моделирование и оптимизационный алгоритм на базе «обучения с подкреплением».

2.2 Описание имитационной модели

Имитационная модель состоит из следующих подсистем (Рис.3):

- Управления временем
- Возникновения угрозы
- Формирования траектории движения объекта-нарушителя
- Выбора станции и запуска БПЛА
- Управления действиями БПЛА
- Выхода из симуляции.

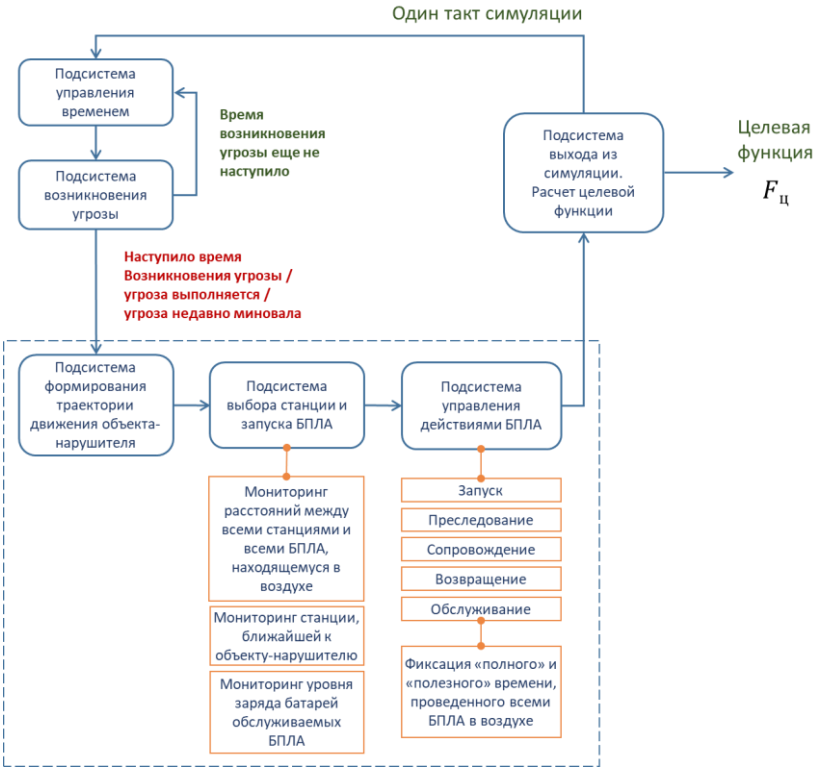


Рис. 3. Блок-схема имитационной модели

Входные данные:

- Δt – число тактов, приходящихся на 1 секунду реального времени. По умолчанию $\Delta t = 1$. Величина $T = 86400\Delta t$ называется длиной «симуляционных суток».

- Прямоугольная дискретная координатная сетка с шагом $\Delta X, \Delta Y$ по осям абсцисс и ординат. $\Delta X, \Delta Y$ измеряются в метрах и привязаны к реальной задаче.
- $\Delta X \cdot k$ – длина стороны квадрата поля зрения камеры.
- $(\tilde{X}_1, \tilde{Y}_1 \dots \tilde{X}_N, \tilde{Y}_N)$ - координаты зоны мониторинга, заданные в виде точек многоугольника.
- $(\tilde{X'}_1, \tilde{Y'}_1 \dots \tilde{X'}_N, \tilde{Y'}_N)$ - координаты внутренних областей, запретных для полетов БПЛА, заданные в виде точек многоугольника.
- $(P_1, \dots, P_N)$ - вероятности проникновения на охраняемую территорию через тот или иной отрезок внешнего периметра.
- Vuav - скорость движения БПЛА, выраженная в количестве метров, которое дрон пролетает за 1 такт симуляции.
- R_c - Скорость зарядки аккумулятора БПЛА,
- Считается, что зарядка аккумулятора происходит равномерно, в соответствии с функцией:

$$ChargeLevel = R_c \cdot t \cdot 100\%, \quad t \le \frac{1}{R_c}$$

$$ChargeLevel = 100\%, \quad t > \frac{1}{R_c}$$

- R_d - Скорость разряда аккумулятора БПЛА. Считается, что разряд аккумулятора происходит равномерно, в соответствии с функцией:

$$ChargeLevel = 100\% - R_d \cdot t \cdot 100\%, \quad t \le \frac{1}{R_d}$$

$$ChargeLevel = 0\%, \quad t > \frac{1}{R_d}$$

Максимальное время полета (в тактах):

$t_{max} = \frac{1}{R_d}$

- K - количество БПЛА, одновременно хранимых на одной станции.
- Vdng - скорость движения объекта, нарушившего охраняемое пространство.
- ν - количество возникающих инцидентов за 86400 тактов симуляции (1 сутки в пересчете на реальное время).

Искомые параметры. Исходя из режима расчета ими могут быть:

- $\{ X_1^{i+1}, Y_1^{i+1} \dots X_n^{i+1}, Y_n^{i+1} \}$ - Искомые координаты расположения наземных станций обслуживания.
- Минимально необходимое число станций *M* и их координаты.

2.3 Работа имитационной модели

Подсистема управления временем осуществляет инкрементальное тактирование алгоритма. Подсистема возникновения угрозы случайным образом, на основе параметра ν, генерирует факт угрозы. Во время первого (*i = 1*) такта каждых новых «симуляционных суток» с помощью генератора псевдослучайных чисел вычисляются номера тактов, в которые произойдут события пересечения периметра:

$$(t_1, t_2 \dots t_\nu)$$

По умолчанию, события пересечения периметра распределены равномерно, на всем временном отрезке (1, T]. Если временной счетчик достигает одного из указанных выше номеров, подсистема помещает в центр одного из отрезков границы охраняемой области точку, характеризующую местоположение объекта нарушителя. При этом выбор отрезка осуществляется на основании заданных ранее вероятностей:

$$(P_1, \dots, P_N)$$

Подсистема формирования траектории управляет движением объекта-нарушителя. Он движется по ломаной прямой:

$$x = k_x t + b_x$$
$$y = k_y t + b_y$$

k_x , k_y , b_x , b_y – изменяются случайным образом через каждые 10 тактов симуляции. В самом начале коэффициенты выбираются таким образом, чтобы объект начал двигаться перпендикулярно стороне многоугольника, внутрь области.

Подсистема выбора станции и запуска БПЛА определяет станции:

С которой взлетает БПЛА для перехвата угрозы.

Куда возвращается БПЛА с разряженной батареей.

Для этого система делит пространство многоугольника с помощью диаграммы Вороного, в которой центрами кластеров выступают координаты станций. Затем, она осуществляет мониторинг всех расстояний, между всеми станциями и находящимися в полете летательными аппаратами, а также расстояние от всех станций до объекта-нарушителя. Зная скорость движения БПЛА, система пересчитывает эти расстояния во время t_0, которое дрон потратит на подлет к цели. Если оставшееся время (с учетом текущего заряда аккумулятора), которое ЛА сможет потратить на сопровождение цели, больше, чем:

$$t_{\text{сопр}} = \ t_{\text{полета}} - 2t_0 = \frac{\text{Уровень заряда в \%}}{100\% R_d} - \ 2t_0 > 0.1t_0$$

то система принимает решение о старте БПЛА с данной станции.

Аналогичным образом происходит выбор станции, на которую дрон отправиться в случае низкого заряда аккумулятора. Этой станцией может быть та, с которой он осуществил взлет, так и любая другая, на которой базируется БПЛА способный подменить его. При этом обмен осуществляется так, чтобы не прекращать сопровождения цели. Поэтому в каждый новый такт симуляции происходит перерасчет станции, куда дрон может вернуться в случае угрозы разряда аккумулятора. Подсистема управления действиями БПЛА рассчитывает направление движения в соответствии с кривой погони. Подсистема выхода из симуляции осуществляет подсчет функции приспособления, а также времени, прошедшего с начала симуляции. В случае достижения заданного времени она возвращает в общий цикл ее значение.

2.4 Оптимизация расположения методом генетического алгоритма

В качестве хромосомы выступает множество $\{ X_1^{i+1}, \ldots X_n^{i+1}, Y_1^{i+1} \ldots Y_n^{i+1} \}$ содержащее искомые координаты расположения наземных станций обслуживания. В самом начале популяция генерируется случайным образом. Выбор родителей происходит методом панмиксии, а скрещивание – точечным методом. К получившимся значениям добавляются небольшие мутации, представляющие собой равномерно распределенные случайные числа, не превышающие 3% от максимальных ширины и длины определенного ранее многоугольника. При этом, получившиеся в результате координаты, не должны выходить за границы многоугольника. Селекция происходит методом рулетки, и затрагивает всю популяцию.

Критерием останова является достижение числа поколений, отпущенных на эволюцию, либо (что важнее) нахождение глобального, либо субоптимального решения. В процессе решения производится несколько «встрясок» системы, в процессе которых поколение генерируется заново. Это позволяет с большей долей уверенности утверждать, что найден именно глобальный максимум. В том случае, если необходимо найти минимальное число станций для мониторинга заданной области, генетический поиск осуществляется для последовательно увеличивающегося числа станций до тех пор, пока не будет достигнуто необходимое значение целевой функции. В силу того, что она отражает эффективность работы системы охраны и мониторинга не рекомендуется использовать значение целевой функции меньшей 0.05, поскольку на сопровождение цели в этом случае будет тратиться менее 5% всего полетного времени БПЛА.

Заключение

В статье рассматривается задача увеличения длительности полёта беспилотных летательных аппаратов (БПЛА) на базе робототехнической станции обслуживания с функцией хранения БПЛА для автоматической перезарядки мультироторных дронов. Подобная станция предназначена для охраны и мониторинга объектов различной инфраструктуры. В статье изучена эффективность применения таких станций, а также решаются задачи определения требуемого количества станций и оптимального их размещения для мониторинга заданной площади. С этой целью предлагается интеллектуальный алгоритм расчета искомых параметров, основанный на комбинировании имитационного моделирования и генетического алгоритма поиска. Входами алгоритма расчета являются координаты зоны мониторинга и внутренних областей, закрытых для полетов БПЛА, вероятности проникновения на охраняемую территорию через внешний периметр, информация о скорости движения БПЛА, его зарядке, количестве БПЛА, одновременно хранимых и обслуживаемых на одной станции, скорости движения объекта-нарушителя, а также данные о частоте возникающих инцидентов. Статья не только предлагает методологию решения задачи оптимального размещения станций и имитационного моделирования с оптимизационным алгоритмом, следующим парадигме «обучения с подкреплением» (reinforcement learning), но и описывает проектируемую конструкцию станции замены батарей. Путем многократного имитационного моделирования процесса вторжения нарушителя в рассматриваемую область и перехвата цели с помощью БПЛА, происходит оптимизация расположения станций обслуживания. В качестве функции оптимизации используется отношение эффективного полетного времени для сопровождения цели к общему времени, проведенному всеми БПЛА в воздухе. На основе заданного критерия оптимизации подбирается минимальное число станций, способных решать требуемую задачу охраны и мониторинга.

Список литературы

[Афанасьев и др., 2015] Афанасьев И.М. и др. Навигация гетерогенной группы роботов (БПЛА и БНР) через лабиринт в 3D симуляторе Gazebo методом вероятностной дорожной карты. Сб. БТС-ИИ, 18-25, 2015.

[Данилов и др., 2016] Данилов И.Ю. и др. Автоматизированные системы для увеличения длительности полета электрических мультикоптеров. Робототехника и искусственный интеллект, 19-24, 2016.

[Нго и др., 2017] Нго К. Т., Соленая О. Я., Ронжин А. Л. Анализ подвижных роботизированных платформ для обслуживания аккумуляторов беспилотных летательных аппаратов. Труды МАИ (95), 11-11, 2017.

[Умников и др., 2018] Умников Е.В., Горковенко В.П. Риски моделирования и симуляции робототехнических комплексов, Известия Института инженерной физики, 1 (47): 82-85, 2018.

[Afanasyev et al., 2015] I. Afanasyev et al. ROS-based SLAM for a Gazebo-simulated mobile robot in image-based 3D model of indoor environment. In Proc. ACIVS. Springer, 273-283, 2015.

[Airobotics, 2018] Airobotics Solution, Automated Industrial drones, https://www.airoboticsdrones.com [Электронный ресурс], 2018.

[Bokovoy et al., 2018] A. Bokovoy, M. Fomin, K. Yakovlev. Implementation of the Pathfinding System for Autonomous Navigation of Mobile Ground Robot. In Proc. ITTMM-WSS, 72-78, 2018.

[Filipenko et al., 2018] Filipenko M., Afanasyev I. Comparison of various slam systems for mobile robot in an indoor environment. In Proc. IS, 2018.

[Gabbasov et al., 2015] B. Gabbasov, I. Danilov, I. Afanasyev, and E. Magid. Toward a human-like biped robot gait: Biomechanical analysis of human locomotion recorded by Kinect-based Motion Capture system, ISMA, 2015.

[Ibragimov et al., 2017] I. Z. Ibragimov et al. Comparison of ROS-based visual slam methods in homogeneous indoor environment. In WPNC, 1-6, 2017.

[Khusainov et al., 2016] Khusainov R. et al. Bipedal robot locomotion modelling with virtual height inverted pendulum and preview control approaches in Simulink environment. In Proc. JRNAL (3): 182-187, 2016.

[Norouzi Ghazbi et al., 2016] S. Norouzi Ghazbi et al. Quadrotors unmanned aerial vehicles: a review. Int. J. on Smart Sensing & Intel. Systems 9.1, 2016.

[Sabirova et al., 2019] A. Sabirova et al., Ground Profile Recovery from Aerial 3D LiDAR-based Maps. In Proc. IEEE FRUCT, Moscow, Russia, 2019.

[Sokolov et al., 2017] Sokolov M. et al. Modelling a crawler-type UGV for urban search and rescue in Gazebo environment. In Proc. ICAROB, 2017.

[Shimchik et al., 2016] I Shimchik, A Sagitov, I Afanasyev, F Matsuno, E Magid. Golf cart prototype development and navigation simulation using ROS and Gazebo. MATEC Web of Conferences, 75, EDP Sciences, 2016.

[Vu et al., 2018] Q. Vu, M. Rakovic, V. Delic & A. Ronzhin. Trends in Development of UAV-UGV Cooperation Approaches in Precision Agriculture. In LNCS (11097), 213-221, Springer, Cham, 2018.

[Zubov et al, 2018] Zubov I. et al. Autonomous Drifting Control in 3D Car Racing Simulator. In Proc. IEEE Intelligent Systems, 2018

УДК 62-519, 001.57, 004.942, 004.946

РАСПОЗНАВАНИЕ РОБОТА В 3D ОБЛАКЕ ТОЧЕК ОТ ОЧКОВ СМЕШАННОЙ РЕАЛЬНОСТИ

В.А. Скворцова (*v.skvortsova@innopolis.university*)
М.А. Останин (*m.ostanin@innopolis.ru*)
И.М. Афанасьев (*i.afanasyev@innopolis.ru*)

Университет Иннополис, Иннополис

Аннотация. Разработка систем интерактивного программирования для промышленных роботов на основе смешанной реальности является сложной задачей робототехники, требующей комплексного подхода. Эта система имеет интуитивно понятный интерфейс и работает на основе очков смешанной реальности Microsoft HoloLens. Поскольку корректная инициализация положения робота в среде влияет на правильную работу интерфейса, важным этапом работы является калибровка системы. При запуске интерфейса Microsoft HoloLens собирает информацию об окружающей среде, создает трехмерное облако точек и получает текущую конфигурацию робота. На основе этого набора данных система инициализирует начальные координаты базы манипулятора. В данной статье описан разработанный алгоритм для инициализации пространственных координат робота путем распознавания и локализации манипулятора в трехмерном пространстве.

Ключевые слова: детектирование роботов, трехмерное облако точек, очки смешанной реальности, Microsoft HoloLens

Введение

В настоящее время увеличивается популярность интерфейсов взаимодействия человека с роботом на основе устройств смешанной реальности, использующих планшеты, смартфоны и специализированные очки. Объединение данных промышленных манипуляторов с восприятием и способностями человека-оператора может привести к появлению инновационных индустриальных приложений, посвященных взаимодействию с роботами и визуализацией их состояния, что, по мнению экспертов, способно повысить эффективность производства и даже увеличить число рабочих мест [Guhl et al., 2017]. В связи с этим, актуальным направлением в робототехнике является создание системы

интуитивно понятного и эффективного взаимодействия человека и робота. В настоящее время Университет Иннополис разрабатывает систему интерактивного программирования промышленных роботов на основе смешанной реальности [Ostanin, 2018]. Эта система имеет интуитивно понятный интерфейс и работает на основе очков смешанной реальности Microsoft HoloLens. Чтобы программное обеспечение работало правильно, оно должно корректно инициализировать положение робота в мировой системе координат, для чего требуется калибровка системы. Калибровка программного обеспечения включает локализацию трехмерной модели робота в облаке точек от MS HoloLens. Поэтому, целью данной работы является разработка быстрого алгоритма для определения локализации промышленного манипулятора в трехмерном облаке точек в соответствии с конфигурацией робота. С этой целью мы сравниваем разные методы обнаружения и локализации объектов в трехмерном облаке точек.

Анализ литературы показывает, что проблема локализации облаков точек актуальна в разных сферах человеческой жизни, таких как детектирование объектов в трехмерном облаке точек на городских улицах [Velizhev et al., 2012], биомеханический анализ походки человека [Gabbasov et al., 2015; Danilov et al., 2016], детектирование рельефа в качестве воздушного картографирования на базе трехмерного лидара [Sabirova, et al., 2019], задачи компьютерного зрения [Nath, 2014] и многие другие. Авторы некоторых статей предлагают различные способы описания моделей для их дальнейшей локализации в пространстве. Так, например, в статье [Velizhev et al., 2012] рассмотрены автоматическая локализация и распознавание объектов в трехмерном облаке точек городских улиц на базе модели неявной формы (ISM)[1], которая распознает объект, выбрав его центральное местоположение. В другом исследовании [Sfikas et al., 2011] предложен метод, основанный на сочетании свойств конформной геометрии и топологической информации на графе, выделяющий контрольные точки объекта и отношения между ними. Другие статьи предлагают методы вписывания модели суперквадратических поверхностей в 3D облако точек на основе алгоритма RANSAC[2] для локализации и оценки позы тела человека [Afanasyev et al., 2012] и распознавания жестов [Afanasyev et al., 2013]. Другие авторы дают свои рекомендации по записи и обработке карт, как это сделано в статье [Rusu et al., 2008], в которой получены трехмерные карты объектов домашней среды (в частности кухни), основанные на PCD[3]. Метод включает статистический анализ, оценку постоянных характеристик

1 от англ. implicit shape models

2 от англ. random sample consensus

3 от англ. point cloud data

гистограмм, подбор и сегментацию среды. Часто вместо описания модели или карты авторы предлагают методы вписывания одного облака точек в другое. Так авторы [Chetverikov et al., 2005] изучают проблему геометрического выравнивания двух зарегистрированных и частично перекрывающихся, зашумленных трехмерных точечных наборов. Для решения этой задачи они предлагают алгоритм последовательного использования подхода наименьших квадратов на всех этапах операции. Похожее решение было предложено в статье [Chen, 2007], использующей интегрированный дескриптор локальной поверхности для распознавания трехмерных объектов. Этот дескриптор характеризуется своим центроидом, типом локальной поверхности и двумерной гистограммой, показывающей частоту появления значений индекса формы против углов между нормалью контрольной точки функции и ее соседями. Однако эти дескрипторы рассчитываются только для характерных точек областей с большим изменением формы. Чтобы ускорить извлечение дескрипторов для большого набора объектов, локальные участки поверхности моделей индексируются в хеш-таблицу. Учитывая набор тестовых локальных наборов поверхности, голоса подаются за модели, содержащие похожие дескрипторы поверхности. На основании соответствия локальных участков поверхности выдвигаются гипотетические модели и выполняется проверка выравнивания моделей с тестовыми данными для наиболее вероятных моделей, встречающихся в сцене с помощью алгоритма итеративной ближайшей точки (ICP)[4]. Другое интересное решение этой проблемы изложено в статье [Tombari, 2010], где авторы предлагают использовать голосование Хафа для обнаружения фигур свободной формы в трехмерном пространстве со значительной степенью окклюзии и беспорядка. Этот метод сопоставляет трехмерные объекты для накопления доказательств присутствия искомых объектов в трехмерном пространстве Хафа.

1 Анализ алгоритмов и моделей для распознавания роботов в 3D облаке точек

В этой главе мы рассматриваем и сравниваем разработанные алгоритмы и модели для распознавания манипуляторов в 3D облаке точек.

1.1 Оценка пространственного положения манипуляторов на основе алгоритма случайной выборки (RANSAC)

Этап предварительной обработки включает в себя понижающую дискретизацию наборов данных модели и сцены, оценку нормалей для сцены и оценку дескрипторов для каждого облака точек. Описание модели

4 от англ. Iterative closest point

основано на примитивах алгоритма согласования случайных выборок (RANSAC), который является итеративным методом оценки параметров математической модели из набора наблюдаемых данных, вычисляющим точки (outliers), не принадлежащие модели. Этот метод считается надежным для установления соответствия между простой трехмерной моделью и облаком, хотя, в нашем случае, нам не удалось получить удачные результаты, в силу сложности модели промышленного манипулятора, который трудно описать простыми моделями геометрических примитивов, такими как цилиндр, куб или шар. Дополнительная проблема заключается в том, что каждая модель в этом случае требует индивидуальной конфигурации. Это требует много ресурсов и не практично с точки зрения реальных решений.

1.2. Локализация роботов на базе алгоритма группирования соответствия (Correspondence Grouping algorithm)

Предварительная обработка данных состоит из следующих этапов: оценка нормалей и описания для каждого облака точек. Обработка представляет собой определение точечных соответствий между моделью и сценой (с помощью k-мерной тройки) и кластеризацию ранее найденных соответствий. Дерево k-d или k-мерное дерево — это структура данных, используемая в информатике для организации некоторого числа точек в пространстве с k измерениями. Это бинарное дерево поиска с другими ограничениями. K-деревья очень полезны для поиска дальности и ближайшего соседа. Для наших целей мы работаем только с облаками точек в трех измерениях, поэтому все наши k-d деревья трехмерные. Каждый уровень дерева k-d разделяет всех дочерних элементов по определенному измерению, используя гиперплоскость, перпендикулярную соответствующей оси. В корне дерева все дочерние элементы разделены на основе первого измерения. Каждый уровень вниз в дереве делится на следующее измерение, возвращаясь к первому измерению, когда все остальные были исчерпаны. Самый эффективный способ построения дерева k-d использует метод разбиения, подобный быстрой сортировке, при котором срединная точка (на полдистанции от начала координат) помещается в корень, а остальные точки – близкие и дальние по дистанции: слева и справа по ветвям соответственно. Затем процедура повторяется для левого и правого поддеревьев, пока последние разделяемые деревья не будут состоять из одного элемента [Yang et al., 2017].

Далее мы используем алгоритм кластеризации на базе преобразования Хафа, выделяющим признаки с поиском объектов в определенном классе фигур с помощью процедуры голосования. Эта процедура выполняется в пространстве параметров (в так называемом аккумуляторе Хафа), в котором локальные максимумы определяют объекты-кандидаты [Tombari,

2010]. В ходе реализации этой методологии выяснилось, что этот алгоритм не может работать с предложенными моделями. Алгоритм мог работать только с данными, полученными очками виртуальной реальности и обработанными вручную. Скорее всего, это происходит по двум причинам: во-первых, очки виртуальной реальности имеют большую ошибку измерения расстояния (наша проверка показала: ошибка измерения расстояния составила 9,72 см), во-вторых, этот алгоритм может работать только с точками, зависящими друг от друга, т.е. если во время выполнения алгоритма не будет достаточного количества зависимых точек, алгоритм предполагает, что в сцене нет модели. Таким образом, эта методология оказалась неработоспособной для локализации роботов.

1.3 Детектирование робота на основе алгоритма итеративной ближайшей точки

В основе следующей методологии лежит алгоритм итеративной ближайшей точки (ICP), этапами которого являются:

1. Связка точек по критерию ближайшего соседа.
2. Оценка параметров преобразования с помощью функции среднеквадратичной стоимости.
3. Преобразования точек с помощью оценочных параметров.
4. Многократные итерации (которые заново связывают точки и т.д).

В этой работе мы рассмотрим две реализации методологий с использованием алгоритма ICP. Идея первой реализации состоит в том, что мы многократно запускаем алгоритм ICP с различными начальными позициями модели на сцене и выбираем однородное преобразование, в котором достигаем наибольшего согласия между моделью и сценой. При этом выполняется следующая последовательность для каждой итерации:

1. Устанавливаем случайную начальную позицию модели на сцене.
2. Запускаем алгоритм ICP.
3. Рассчитываем точность совпадения точек сцены и модели.
4. Если точность максимальна, то получаем наилучшее однородное преобразование.

Основным преимуществом такой реализации является то, что, многократно повторяя алгоритм при разных начальных условиях, мы уменьшаем вероятность достижения локального минимума. Этот метод оказался очень надежным по сравнению с другими алгоритмами. Из 8 тестов он сработал все 8 раз, имея 100 итераций в каждой попытке. На Рис. 1 показан результат работы этого алгоритма. Основным недостатком алгоритма является скорость его выполнения. В среднем расчеты занимают 53 минуты для каждого теста.

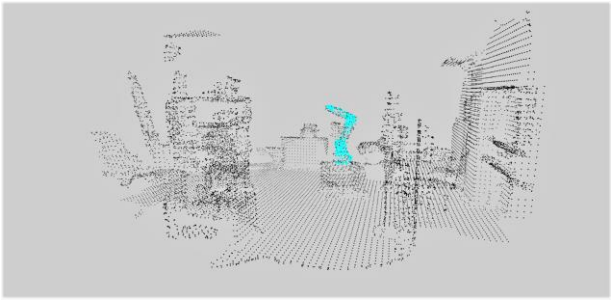


Рис 1. Детектирование модели манипулятора KUKA Agilus на основе алгоритма итеративной ближайшей точки со случайным положением модели

Вторая реализация основана на том, что мы пытаемся определить количество объектов в сцене с помощью кластерного анализа. В этом случае пользователь должен самостоятельно указать количество наблюдаемых в сцене объектов. Тогда мы сравниваем объем кластеризованных объектов с объемом модели и выбираем наиболее подходящее облако. Затем мы запускаем алгоритм ICP, который работает быстрее, поскольку размер выбранного облака будет на порядок меньше, чем размер облака сцены. Однако, эта реализация оказалась менее точной, чем предыдущая. Из 8 тестов метод правильно определил однородную трансформацию только в 3 случаях. Результат удачной работы этого алгоритма показаны на Рис. 2. Это связано с тем, что даже несмотря на уменьшение размера сцены, вероятность попадания в локальный минимум остается высокой. Также была проблема с определением количества объектов в сцене. Однако этот метод работает быстрее, демонстрируя среднее время выполнения теста – 35 секунд.

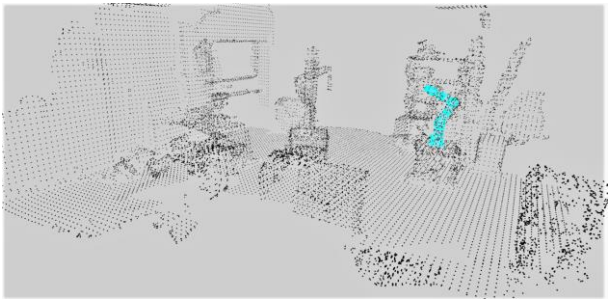


Рис 2. Детектирование модели промышленного манипулятора KUKA IIWA на основе алгоритма итеративной ближайшей точки с кластеризацией сцены

Таблица. 1. Сравнение разработанных алгоритмических решений для распознавания модели манипуляторов в 3D облаке точек.

Название алгоритма	Применимость к решению поставленной задачи	Точность	Время обработки
Оценка положения манипуляторов на основе алгоритма случайной выборки	Нет	-	-
Локализация роботов на базе алгоритма соответствующей группировки	Нет	-	-
Детектирование робота на основе алгоритма итеративной ближайшей точки со случайным положением модели	Да	0.03 м	53 минуты
Детектирование робота на основе алгоритма итеративной ближайшей точки с кластеризацией сцены	Да	от 0.03 до 2.5 м	35 секунд
Локализация роботов в 3D облаке точек на базе алгоритма плотности шума (DBSCAN)	Да	от 0.04 м	23 секунды

2 Разработка алгоритма локализации роботов в 3D облаке точек на базе алгоритма плотности шума (DBSCAN)

Таким образом, после имплементации и анализа работы всех методологий (описанных в Разделе 1) мы создали наш собственный алгоритм, в котором учли слабые стороны алгоритмов на базе ICP и использовали их преимущества. Блок-схема нашего алгоритма приведена на Рис. 3. Предварительные обработки облаков точек сцены и модели отличаются. Внутренние точки были обнаружены в большинстве модельных облаков точек. Это вносит большие ошибки в работу любого алгоритма распознавания. Мы разделяем облако точек модели на слои вдоль одной из осей координат и удаляем внутренние точки, используя алгоритм Джарвиса, чтобы построить минимальную выпуклую оболочку

для каждого слоя. Затем, мы удаляем пол в облаке точек сцены, чтобы упростить процедуру кластеризации. Мы используем алгоритм RANSAC для инициализации плоскостей. После этого убираем плоскость с минимальной z-координатой. Следующим этапом предварительной обработки сцены является кластеризация точек сцены в объектах. Для этой цели мы используем пространственную кластеризацию на основе алгоритма плотности шума (DBSCAN) [Schubert et al., 2017], который представляет собой алгоритм кластеризации на основе плотности: при заданном наборе точек в некотором пространстве он группирует точки, которые тесно упакованы вместе (точки с множеством соседей), помечая выпадающие точки, которые лежат одни в областях с низкой плотностью.

Рис. 3 Блок-схема предложенного алгоритма.

После этапа предварительной обработки мы измеряем объем модели и каждого кластерного объекта. Для этого мы подбираем наиболее похожие объекты по объему. Установив начальное положение модели для объекта (связывая средний вектор каждого из облаков точек) и запустив ICP-алгоритм, мы повторяем этот шаг для каждого подходящего объекта с точки зрения объема и записываем значение ошибки для каждого случая. После чего выбираем преобразование, при котором ошибка была наименьшей. Результат работы показал, что алгоритм работает правильно как для разреженных, так и для плотных облаков точек (Рис. 4). Среднее

время работы алгоритма составляет 23 секунды, что быстрее, чем работа других методологий. Из 8 тестов он правильно определил трансформацию в 7 случаях с ошибкой в 0.04 м. Во время работы алгоритм может сообщать об ошибке, если в сцене присутствуют два робота одинакового размера в одинаковой конфигурации.

Результаты работы разных алгоритмов для распознавания модели манипуляторов в 3D облаке точек, включая предложенный нами метод на базе алгоритма плотности шума (DBSCAN), представлены в Таблице 1.

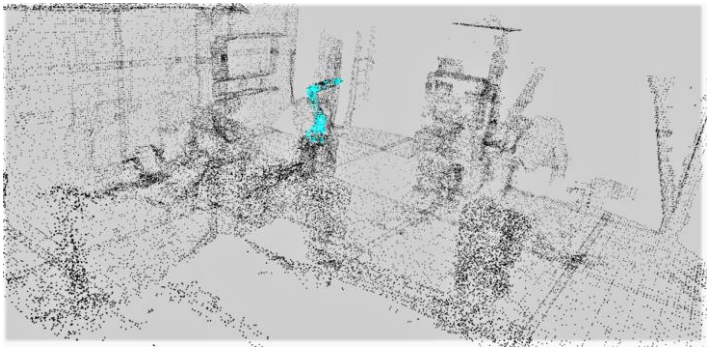


Рис. 4 Результат работы алгоритма на плотном облаке точек.

Заключение

В статье был предложен разработанный быстрый и точный алгоритм локализации 3D модели промышленного манипулятора в сцене, которая была получена с помощью очков смешанной реальности Microsoft HoloLens. На основе 3D облаков точек внешней среды и модели алгоритм инициализировал координаты базы манипулятора. Этот проект решает проблему с калибровкой системы интерактивного программирования промышленных манипуляторов, используя очки смешанной реальности. Мы достигли быстрой работы алгоритма (порядка 20 секунд) и точности в 0.04 м. Алгоритм надежно работает в ситуациях, когда в сцене присутствует один детектируемый робот необходимой конфигурации, в противном случае точность алгоритма не гарантируется.

Список литературы

[Afanasyev et al., 2012] I. Afanasyev et al. 3D Human Body Pose Estimation by Superquadrics. VISAPP (2): 294–302, 2012.

[Afanasyev et al., 2013] I. Afanasyev and M. De Cecco. 3D Gesture Recognition by Superquadrics. VISAPP (2): 429–433, 2013.

[Chen, 2007] H. Chen and B. Bhanu. 3D free-form object recognition in range images using local surface patches. Pattern Recognit. Lett., 28(10), 2007.

[Chetverikov et al., 2005] D. Chetverikov, D. Stepanov, and P. Krsek. Robust Euclidean alignment of 3D point sets: The trimmed iterative closest point algorithm. Image and Vision Computing, 23(3): 299–309, 2005.

[Danilov et al., 2016] I. Danilov, B. Gabbasov, I. Afanasyev, and E. Magid, ZMP Trajectory from Human Body Locomotion Dynamics Evaluated by Kinect-based Motion Capture System. In VISAPP, pp. 162-168, 2016.

[**Gabbasov et al., 2015**] B. Gabbasov, I. Danilov, I. Afanasyev, and E. Magid Toward a human-like biped robot gait: Biomechanical analysis of human locomotion recorded by Kinect-based Motion Capture system, ISMA, 2015.

[Guhl et al., 2017] J. Guhl, S. Tung, J. Kruger, Concept and architecture for programming industrial robots using augmented reality with mobile devices like Microsoft HoloLens. In IEEE ETFA, pp. 1-4, 2017.

[Nath, 2014] V. Nath and S. E. Levinson. Computer vision. Springer Briefs in Computer Science, (9783319056050):33–38, 2014.

[Ostanin, 2018] M. Ostanin and A. Klimchik. Interactive Robot Programing Using Mixed Reality. IFAC Symposium on Robot Control, 2018.

[Pomelau et al., 2015] F. Pomerleau, F. Colas, and R. Siegwart. A Review of Point Cloud Registration Algorithms for Mobile Robotics. Foundations and Trends in Robotics, 4(1):1–104, 2015.

[Rusu et al., 2008] R. B. Rusu et al Towards 3D Point cloud based object maps for household environments. Robot Auton Syst, 56(11):927–941, 2008.

[Sabirova et al., 2019] A. Sabirova, et al., Ground Profile Recovery from Aerial 3D LiDAR-based Maps. In FRUCT, arXiv:1903.11097, 2019.

[Schubert et al., 2017] Schubert Erich et al. DBSCAN Revisited, Revisited: Why and How You Should (Still) Use DBSCAN. ACM TODS, 10197, 2017.

[Sfikas, 2011] K. Sfikas, et al. ConTopo: Non-rigid 3D object retrieval using topological information guided by conformal factors. EG 3DOR, 2011.

[Skibbe, 2012] H. Skibbe, et al. Fast rotation invariant 3D feature computation utilizing efficient local neighborhood operators. IEEE TPAMI, 34(8), 2012.

[Tombari, 2010] F. Tombari and L. Di Stefano. Object recognition in 3D scenes with occlusions and clutter by Hough voting. PSIVT, 2010.

[Velizhev et al., 2012] A. Velizhev, et al. Implicit shape models for object detection in 3D point cloud. ISPRS Annals:179–184, 2012.

[Wohlhart, 2015] P. Wohlhart and V. Lepetit. Learning descriptors for object recognition and 3D pose estimation. IEEE CVPR, 3109–3118, 2015.

[Yang et al., 2017] J. Yang, K. Xian, Y. Xiao, Z. Cao. Performance evaluation of 3D correspondence grouping algorithms. In IEEE 3DV, 467-476, 2017.

УДК 007.52, 519.878, 519.1, 004.942, 006.72

ПЕРЕНОС ПОДХОДА МАШИННОГО ОБУЧЕНИЯ С ПОДКРЕПЛЕНИЕМ С СИМУЛЯЦИОННОЙ МОДЕЛИ НА МОБИЛЬНОГО РОБОТА

А.Г. Сагитов (*sagitov@it.kfu.ru*)
Казанский Федеральный Университет, Институт Информационных технологий и интеллектуальных систем, Лаборатория интеллектуальных робототехнических систем, Казань

Tetsuto Takano (takanotetuto@gmail.com)
Университет Канадзавы, Канадзава, Япония

Shohei Muto (nsmkaa@gmail.com)
Университет Канадзавы, Канадзава, Япония

Р.О. Лавренов (lavrenov@it.kfu.ru)
Казанский Федеральный Университет, Институт Информационных технологий и интеллектуальных систем, Лаборатория интеллектуальных робототехнических систем, Казань

Аннотация. Обучение с подкреплением, как один из способов машинного обучения, показывает многообещающие результаты при его интеграции в различные робототехнические алгоритмы. Но для того, чтобы добиться оптимального поведения робота, требуется значительное количество времени и ресурсов. Используя виртуальные эксперименты, возможно значительно ускорить и улучшить производительность алгоритмов. Мы внедрили подход обучения с подкреплением для алгоритма локализации и картографирования, применяемого на мобильном роботе. Алгоритм был обучен в симуляционной среде Gazebo и перенесен на реального робота. В публикации показана целесообразность использования симуляции для обучения алгоритмов, применяемых мобильными роботами.[1]

Ключевые слова: алгоритм, мобильный робот, машинное обучение, моделирование, Gazebo.

[1] Работа выполнена при финансовой поддержке РФФИ (проект № 19-58-70002).

Введение

Обучение с подкреплением (англ. reinforcement learning) — это метод машинного обучения, в основе которого лежит процесс корректировки модели поведения агента при взаимодействии с некоторой средой на основе функционала выигрыша (оценка действий агента). Процесс обучения, таким образом, является процедурой нахождения таких значений параметров модели поведения, которые обеспечивающих максимум выигрыша. Метод основан на принципах, аналогичных преподаванию в школе: сдача тестов и экзаменов, с получением оценок (положительных и отрицательных) для последующей работы над ошибками. В робототехнике такой метод является одним из способов проектирования и реализаций сложных моделей поведения, особенно предполагающих взаимодействия с неопределенной или динамичной средой.

Первоначальной целью исследования было создание колесного мобильного робота, выполняющего автономное исследование окружающей среды, на основе алгоритма обучения с подкреплением. Основным фактором, который должен быть учтён при использовании метода обучения с подкреплением — это общее количество времени, которое необходимо агенту на оптимизацию параметров поведения. Реальные эксперименты в таком случае являются достаточно медленными и затратными, и не исключают вероятности поломки агента или части окружения. Одним из возможных решений этой проблемы является имитационное моделирование эксперимента. В процессе обучения на модельных экспериментах алгоритм может быть апробирован в реальных условиях с последующей корректировкой параметров моделируемой среды.

Одним из основных инструментов разработки и верификации алгоритмов машинного обучения с подкреплением является система OpenAI Gym [Brockman, 2016]. Система представляет набор инструментов и библиотек для отладки и тестирования разработанных алгоритмов для различных окружений. В работе [Zamora et al., 2016] авторы интегрировали инструменты OpenAI Gym в Робототехническую Операционную Систему ROS и систему имитационного моделирования Gazebo, упростив тем самым интеграцию алгоритмов обучения в существующие РТС. Была также добавлена возможным отладки существующих алгоритмов на роботах в реальных условиях.

В рамках этой работы сделанный нами мобильный робот был интегрирован в среду ROS\Gazebo для решения задачи автономного исследования окружения. Разработанные интерфейсы управления были унифицированы для реального и виртуального робота. Таким образом, после завершения обучения в симуляции полученная стратегия сразу

может быть перенесена на реального робота для апробации. Обучение робота произведено на основе алгоритма Q-Learning, используя функционал вознаграждения робота, поощряющий исследование ранее неисследованных областей и штрафующий за столкновения с препятствиями окружающей среды и пассивность.

1 Существующие подходы

Математическая концепция обучения с подкреплением появилась в результате поиска оптимального управления Марковским процессом принятия решений в работах Беллмана [Bellman, 1957]. Дальнейшие принципы обучения с подкреплением были развиты в работах [Sutton, 1984], [Watkins, 1989]. В последние годы обучение с подкреплением стало одним из важных методов в робототехнике [Kober et al., 2013]. Этот метод может быть применен для передвижения робототехнических систем (РТС) [Kohl et al., 2004], [Endo et al., 2008], манипулирования объектами [Peters et al., 2008], [Theodorou et al., 2010], [Peters et al., 2010], [Kalakrishnan et al., 2011], планирования движения автономных летающих устройств [Abbeel et al., 2006] и других задач.

Комбинирование подхода обучения с подкреплением на основе нейронных сетей показало значительный потенциал при выполнении сложных взаимодействий, например, для управления в реальном времени семиосевыми манипуляторами [Pastor et al., 2009], [Schulman et al., 2015], [Levine et al., 2016]. Использование больших и глубинных нейронных сетей (англ. deep neural networks) позволило роботам освоить сложные манипуляции с минимальной корректировкой траекторий, ставя вопрос о применение подобного алгоритм к решению произвольных задач управления [Deisenroth et al., 2011], [Moldovan et al., 2015].

В исследовании [Sadeghi et al., 2018] сложная нейронная сеть (комбинация сверточной нейронной сети с моделью долго-краткосрочной памяти) успешно выполнила автокалибровку суставов на основе сохраненной истории произведенных действий и показаний датчиков. Помимо этого, обучив подобную модель на наборе синтетических (смоделированных) выборок, состоящих из набора конечных положений с рассчитанными траекториями, модель была использована для управления роботом манипулятором для достижений различных конечных положений в произвольных координатных системах.

2 Моделирование объекта и процесса обучения

Для организации процесса обучения виртуального робота в виртуальной среде мы использовали среду выполнения OpenAI gym-gazebo. Среда gym-gazebo является комбинацией системы машинного

обучения OpenAI Gym, системы управления ROS и среды физического моделирования Gazebo. OpenAI gym предоставляет набор интерфейсов для реализации, тестирования и отладки алгоритма обучения управляющего поведением реального и виртуального робота, в том числе и интерфейс взаимодействия со средой Gazebo [Afanasyev et al., 2015].

2.1 Обучаемый робот

В качестве обучаемого робота был выбран специально разработанный мобильный робот, управляемый с помощью установленного микроконтроллера Arduino Uno (Рис. 1). Обучаемый робот представляет собой мобильный четырехколесный робот с четырьмя двигателями постоянного тока (по одному на колесо, по два двигателя с каждой стороны). Каждое колесо оборудовано пружинной амортизацией. Для каждой оси был реализован программный контроллер скорости на основе библиотеки *ros_control* системы ROS. Робот реализует модель дифференциального рулевого управления при выполнении поворотов и полных разворотов на месте, создавая разницу в скоростях колес с левой или правой сторон. Для оценки окружения робота был использован лазерный дальномер Hokuyo LIDAR (UTM-30LX), установленный на верхней передней части робота.

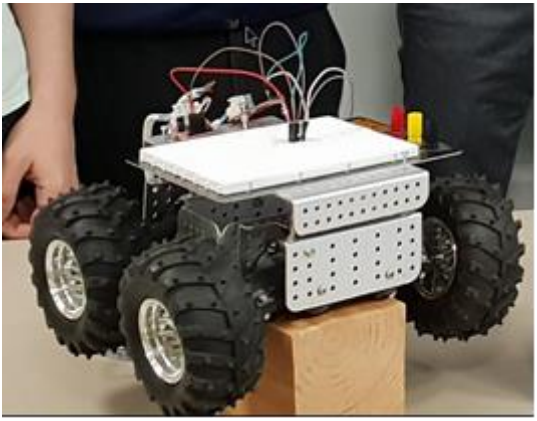


Рис. 1. Обучаемый робот на контроллере Arduino Uno.

Протоколом связи робота с ROS является цифровой интерфейс UART, реализованный на основе библиотеки *rosserial* системы ROS, с предоставлением доступа к коммуникационной подсистеме ROS, с интеграцией координатных систем, определенных в разных системах отсчёта. Также была выполнена синхронизация работы двигателя и

датчика с системным временем ROS. Использованные протоколы *rosserial* были адаптированы с помощью написания оберточных функций над стандартными типами сообщений ROS с последующим мультиплексированием несколько сообщений с микроконтроллера.

Виртуальное представление обучаемого робота было создано в среде Gazebo с реализацией контроллеров управления, моделей датчиков и интерфейсов модели с адаптацией под поведение реального робота, (Рис. 2), используя подход, примененный нами ранее для моделирования гусеничного робота [Sokolov et al., 2016], [Лавренов и др., 2018].

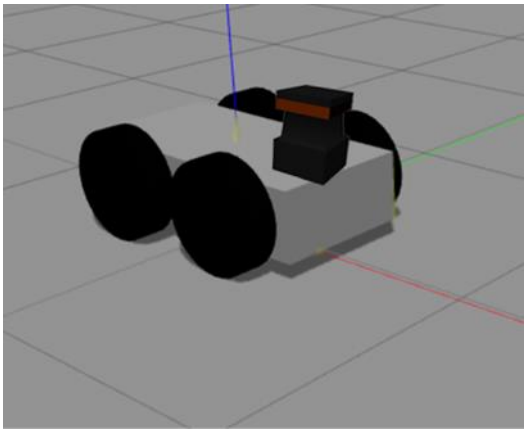


Рис. 2. Виртуальный робот в Gazebo.

2.2. Алгоритм обучения

Для организации процесса обучения РТС процедуре исследования окружающей среды был использован широко распространенный многошаговый метод обучения с подкреплением Q-Learning [Watkins, 1989]. Поведение робота в модели определяется набором правил, определяющих возможные действия в различных обстоятельствах, отношение между действиями и вознаграждениями при их выполнении являются вероятностными переходами.

Процесс исследования роботом окружающей среды был представлен как модель конечного Марковского процесса принятия решений (англ. FMDP - finite Markov decision process). Используя алгоритм Q-learning, был произведен поиск оптимальных параметров модели, определенных максимумом ожидаемого значения выигрыша (в нашем случае отношения исследованной области к общей) в каждом из действий робота. Начиная с некоторой начальной точки в пространстве принятия решений, алгоритм Q-learning позволил найти оптимальные параметры для модели поведения

(в виде формализованного FMDP). В качестве функционала выигрыша было принято отношение меры исследованного пространства к общему. Вычисление отношения производилось на основе сравнения карты, построенной роботом на основе показаний лидара, и модельной карты лабиринта.

Робот строит свою карту используя исключительно показания лидара. Для каждой модельной среды определена модельная карта, представляющая из себя желаемое состояние – полностью исследованная карта.

Обучаемый робот — это кортеж состояний S и набора возможных действий из этих состояний A. Состояние робота S представляет собой набор текущих координат робота, простроенной на основе показаний лидара карты и текущей оценке. В случае выбора действия $\mathbf{a} \in A$, робот выполняет переходы из текущего состояния в состояние, определенное действием **a.** После каждого перехода, робот подсчитывает суммарное вознаграждение на основе функционала. По завершению определенного количества шагов, или в случае, когда робот застревает и не может двигаться дальше, проводится расчет окончательного вознаграждения.

Робот изменяет параметры переходов для максимизации суммарного вознаграждения, в результате поиска максимума прогнозируемого вознаграждения в будущих состояниях. Общее потенциальное суммарное вознаграждение определяется как взвешенная сумма ожидаемых значений вознаграждений за все будущие действия, начиная с текущего состояния, и является целевой функцией для поиска оптимальных параметров поведения.

После выполнения действия целевая функция обновляется:

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha \, |r_t + \gamma \max Q(s_{t+1},a_t) - Q(s_t,a_t)| \qquad (1)$$

где r_t – вознаграждение за шаг t, параметры α и γ являются гиперпараметрами алгоритма обучения, α - скорость обучения, определяющая влияние новой полученной информации, γ коэффициент дисконтирования, определяющий выбор максимума будущего или текущего вознаграждения.

Для нашего обучаемого робота определены четыре возможных действия в любом состоянии: движение вперед (одновременное вращение обеих осей со скоростью +0,2 м/с), левый поворот (левая ось -0,1 м/с, правая ось +0,1 м/с), правый поворот (левая ось +0,1 м/с, правый ось -0,1 м/с) и движение задним ходом (обе оси -0,2 м/с). Вознаграждение робота штрафовалось в случаях касания роботом препятствий. Шаг по времени выбран размером в 1 секунду, по выполнению выбранной формы

движения, на основе данных лидара карта робота достраивалась и рассчитывалось текущее вознаграждение.

2.3 Среда исследования

В качестве тестовой среды была выбрана одна из стандартных симуляций Gazebo: GazeboCircuit2TurtlebotLidar-v0. Эта среда состоит из закольцованного коридора для робота с пятью правыми поворотами и одним левым поворотом (Рис. 3). Был использован датчик LIDAR для локализации и картографирования окружающей среды. Больше никакой информации, с помощью которой можно было бы судить о положении робота, не использовалось.

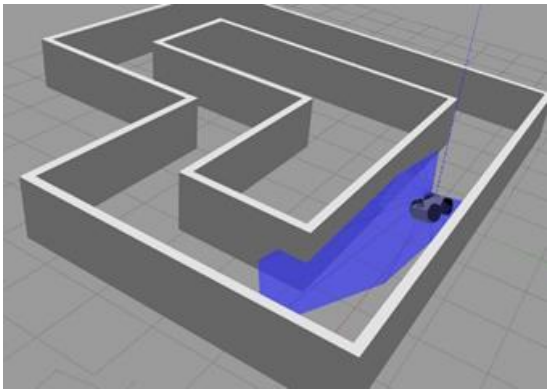


Рис. 3. Среда для тестирования в ROS/Gazebo.

3 Результаты

Использование смоделированного робота в виртуальной среде позволило обучить алгоритм примерно в 10 раз быстрее, чем с помощью обучения в режиме реального времени. Таким образом виртуальный робот может выполнять в десять раз больше обучающих экспериментов, чем настоящий робот. Так, более пяти тысяч запусков алгоритма исследования виртуальной среды были завершены в течение 6 часов.

Рис. 4. Суммарное вознаграждение в ходе запусков алгоритма.

На рисунке 4 продемонстрировано, что средняя общая награда постепенно увеличилась, следовательно, алгоритм исследования среды значительно улучшился с течением времени. Даже с тренировками, выполненными полностью в симуляторе, были получены стратегии исследования, которые хорошо работают на физическом роботе. Мы связываем успешный переход, в первую очередь, с унифицированным интерфейсом управления между реальным и виртуальным роботом. Однако, были замечены несколько факторов, которые способствовали различию между симуляцией и реальностью во время выполнения алгоритма. Реальный робот, в отличии от виртуального, демонстрирует проскальзывание колеса в поперечном и продольном направлениях, накапливает неточности скорости вращения колеса и более склонен к столкновениям при движении возле стен.

4 Планы дальнейшей работы

В перспективе мы заинтересованы проверить метод на более сложных роботах с использованием большего набора различных сенсоров. Для дальнейшего улучшения процесса обучения мы планируем распараллелить процесс расчетов и реализовать автоматические рекомендации из результирующей стратегии. Также будет проведена дальнейшая диверсификация среды и внедрены дополнительные инструменты для расчета показателей производительности для различных алгоритмов.

Заключение

Обучение с подкреплением играет важную роль в растущей области машинного обучения. Чтобы преодолеть трудности, связанные с обучением, использование робототехнических симуляторов, таких как Gazebo, позволяет существенно снизить стоимость и повысить скорость разработки. В данной работе было продемонстрировано, что алгоритмы обучения с подкреплением способны изучать сложные навыки исследования с нуля и без целенаправленно разработанных траекторий. Тем не менее, наш метод имеет ряд существенных ограничений, включая упрощенные модели робота и тестовой среды.

Список литературы

[Brockman, 2016] G. Brockman, et.al., OpenAI gym (arXiv preprint, arXiv:1606.01540, 2016).

[Zamora et al., 2016] I. Zamora, et.al. Extending the OpenAI Gym for robotics: a toolkit for reinforcement learning using ROS and Gazebo (arXiv preprint arXiv:1608.05742, 2016).

[Bellman, 1957] R. Bellman, A Markovian decision process, in *Journal of Mathematics and Mechanics* (1957) pp. 679-684.

[Sutton, 1984] R.S. Sutton, Temporal credit assignment in reinforcement learning, (PhD Thesis, University of Massachusetts, Amherst, MA., 1984).

[Watkins, 1989] C.J.C.H. Watkins, Learning from Delayed Rewards (Ph.D. thesis, Cambridge University, 1989).

[Kober et al., 2013] J.Kober, J.A. Bagnell, J. Peters, Reinforcement learning in robotics: A survey, in *The Int. J. of Robotics Research*, 32, (2013), pp. 1238-1274.

[Kohl et al., 2004] N. Kohl and P. Stone, Policy gradient reinforcement learning for fast quadrupedal locomotion, in *Int. Conf. on Robotics and Automation* 3 (2004), pp. 2619-2624.

[Endo et al., 2008] G. Endo, J. Morimoto, T. Matsubara, J. Nakanishi, and G. Cheng, Learning CPG-based biped locomotion with a policy gradient method: Application to a humanoid robot, in *Int. J. of Robotic Research*, **27**(2) (2008), pp. 213–228.

[Peters et al., 2008] J. Peters and S. Schaal, Reinforcement learning of motor skills with policy gradients, in *Neural Networks*, **21**(4) (2008), pp. 682–697.

[Theodorou et al., 2010] E. Theodorou, J. Buchli, and S. Schaal, Reinforcement learning of motor skills in high dimensions, in *Int. Conf. on Robotics and Automation* (2010) pp. 2397-2403.

[Peters et al., 2010] J. Peters, K. Mulling, and Y. Altun, Relative entropy policy search, in *AAAI Conference on Artificial Intelligence* (2010), pp. 1607-1612.

[Kalakrishnan et al., 2011] M. Kalakrishnan, L. Righetti, P. Pastor, and S. Schaal, Learning force control policies for compliant manipulation, in *Int. Conf. on Intelligent Robots and Systems* (2011), pp. 4639-4644.

[Abbeel et al., 2006] P. Abbeel, A. Coates, M. Quigley, and A. Ng, An application of reinforcement learning to aerobatic helicopter flight, in *Advances in Neural Information Processing Systems* (2006) pp. 1-8.

[Pastor et al., 2009] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal, Learning and generalization of motor skills by learning from demonstration, in *Int. Conf. on Robotics and Automation* (2009), pp. 763-768.

[Schulman et al., 2015] J. Schulman, S. Levine, P. Moritz, M. Jordan, and P. Abbeel, Trust region policy optimization, in *Int. Conf. on Machine Learning* (2015), pp. 1889-1897.

[Levine et al., 2016] S. Levine, C. Finn, T. Darrell, and P. Abbeel, End-to-end training of deep visuomotor policies, *J. of Machine Learning Research*, **17**(1) (2016), pp. 1334-1373.

[Deisenroth et al., 2011] M. Deisenroth and C. Rasmussen, PILCO: a model-based and data efficient approach to policy search, in *Int. Conf. on Machine Learning* (2011), pp. 465-472.

[Moldovan et al., 2015] T. Moldovan, S. Levine, M. Jordan, and S. Abbeel, Optimism-driven exploration for nonlinear systems, in *Int. Conf. on Robotics and Automation* (2015), pp. 3239-3246.

[Sadeghi et al., 2018] F. Sadeghi, A. Toshev, E. Jang and S. Levine, Sim2Real Viewpoint Invariant Visual Servoing by Recurrent Control, in *IEEE Conf. on Computer Vision and Pattern Recognition* (2018), pp. 4691-4699.

[Afanasyev et al., 2015] I., Afanasyev, A., Sagitov, E., Magid. ROS-based SLAM for a Gazebo-simulated mobile robot in image-based 3D model of indoor environment. In *Int. Conf. on Advanced Concepts for Intelligent Vision Systems*, (Springer, Cham, 2015), pp. 273-283.

[Sokolov et al., 2016] Sokolov M. et al. 3D modelling and simulation of a crawler robot in ROS/Gazebo //*Proceedings of the 4th International Conference on Control, Mechatronics and Automation*. – ACM, 2016. – С. 61-65.

[Лавренов и др., 2018] Лавренов Р. О. и др., Робот "Сервосила Инженер": Разработка сервера передачи видеопотока и интерфейса управления под фреймворк ROS // *Известия ЮФУ. Технические науки*. – 2018. – №. 1. – С. 294-309.

УДК 681.786

ПИЛОТНЫЕ ВИРТУАЛЬНЫЕ ЭКСПЕРИМЕНТЫ ПО СРАВНЕНИЮ СИСТЕМ КООРДИНАТНЫХ МЕТОК ARUCO И APRILTAG НА УСТОЙЧИВОСТЬ К ВРАЩЕНИЮ

А.А. Закиев, К.С. Шабалина, Т.Г. Цой, Е.А. Магид
(zaufar@it.kfu.ru, ks.shabalina@it.kfu.ru, tt@it.kfu.ru, magid@it.kfu.ru)
Казанский федеральный университет, Институт Информационных технологий и интеллектуальных систем, Лаборатория интеллектуальных робототехнических систем, Казань

Аннотация. В настоящее время, ввиду большого количества различных систем координатных меток научное сообщество и промышленность сталкиваются с трудностями при выборе оптимальной системы координатных меток для выполнения определенной задачи. В результате, каждая группа выбирает систему меток и идентификатор метки интуитивно или на основе своего предыдущего опыта. В данной работе представлен дизайн экспериментов и пилотные виртуальные эксперименты, позволяющие сравнить результаты распознавания различных систем координатных меток. Эксперименты были разработаны для оценки устойчивости систем координатных меток к вращению относительно различных осей в трехмерном пространстве. В виртуальных экспериментах исключено влияние внешней среды, включая условия освещения, шумы сенсора, неточности движений метки и другие. Эксперименты проводились в среде ROS/Gazebo с использованием двух систем координатных меток: ArUco и AprilTag. Для сбора статистически значимого количества данных было проведено и проанализировано более 300,000 виртуальных экспериментов.[1]

Ключевые слова: робототехника, калибровка, техническое зрение, система координатных меток, виртуальные эксперименты, ROS, Gazebo.

[1] Работа выполнена при финансовой поддержке РФФИ (проект № 18-58-45017).

Введение

Системы координатных меток (СКМ) — это системы плоских графических изображений, спроектированных для определения и распознавания соответствующими алгоритмами технического зрения. СКМ широко используются в физике, медицине, робототехнике, дополненной реальности, метрологии, и других областях. Широкий перечень робототехнических задач, включая навигацию [Kuriya et al., 2015], локализацию [Dhiman et al., 2013], построение карты [Olson, 2011] и калибровку камеры [Fiala, 2005b], использует СКМ в качестве основного элемента. Нашей долгосрочной целью является автономная калибровка камер нескольких российских роботов, включая антропоморфного робота AR-601M (Рис. 1), с применением СКМ в автономном режиме. Современные СКМ были разработаны для различных целей, поэтому у каждой из них имеются свои преимущества и недостатки. Выбор подходящей СКМ среди существующего многообразия возможных вариантов требует сравнения систем по различным критериям, уделяя особое внимание именно тем критериям, которые являются важными для выполнения конкретной поставленной задачи. Нашей целью является автономная калибровка камер мобильного робота, а координатные метки располагаются на теле робота. Например, для AR-601M они располагаются на тыльной стороне ладони манипулятора, и робот, вращая манипулятор перед объективами бортовых камер, наблюдает за меткой для оценки и программного устранения искажения изображений с камеры. Для этой задачи СКМ должна быть устойчива к вращению манипулятора и частичным перекрытиям, которые возникают в результате перекрытия метки различными объектами (например, частями самого робота при движении манипулятора в процессе калибровки). Другими словами, используемая СКМ должна иметь максимальные углы инвариантности распознавания меток к изменению позиции камеры – отдельно для разных осей вращения.

Существующие на момент написания этой статьи исследования по сравнению разных СКМ не отличаются проработанным дизайном экспериментов. К их недостаткам относятся произвольный выбор ракурсов съемки при распознавании [Fiala, 2005b], отсутствие системного подхода к осуществлению перекрытия метки (например, использование пальца или руки для перекрытия) [Garrido-Jurado et al., 2014], а также сравнение авторами старого и нового алгоритмов распознавания одной и той же СКМ [Wang et al., 2016].

В наших предыдущих научных исследованиях [Sagitov et al., 2017] [Shabalina et al., 2017] мы изучали СКМ AprilTag, ARTag и CALTag [Atcheson et al., 2010]. Изучение велось путем сравнения этих систем по

критериям устойчивости к вращениям, систематическим окклюзиям и произвольным перекрытиям. Данные эксперименты проводились вручную сначала с использованием веб-камеры Genius FaceCam 1000X для получения данных об актуальности применения СКМ в условиях использования недорогого оборудования. Далее эксперименты проводились на антропоморфном роботе AR-601M, используя его моно-камеру Basler acA640-90gc. Анализ результатов эксперимента показал, что AprilTag и ARTag устойчивы к вращениям метки, однако, очень чувствительны к перекрытиям краев метки. Это объясняется принципом работы алгоритма распознавания: первый этап заключается в поиске границ потенциальных меток и, если они не обнаруживаются, то весь процесс распознавания останавливается. С другой стороны, CALTag продемонстрировал стабильно высокий процент распознавания на различных углах наклона и различных перекрытиях.

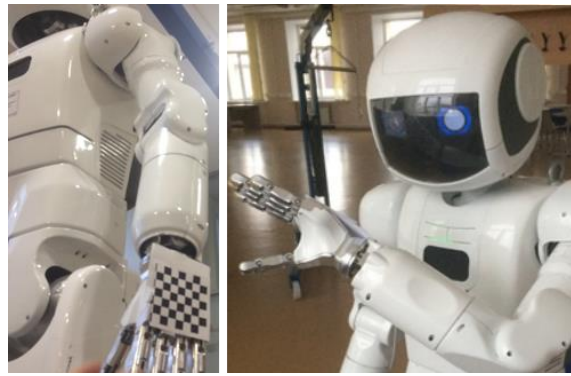


Рис. 1. Расположение координатной метки на манипуляторе робота (слева); робот наблюдает за меткой в процессе детектирования и распознания метки (справа).

Однако проведение экспериментов вручную имеет ряд очевидных недостатков, которые не позволяют на практике повторить эксперименты и проверить их результаты:

- Требуется большое количество времени для осуществления экспериментов, так как требуются многочисленные итерации для сбора статистически значимого количества данных.
- Сложность контроля чистоты экспериментов. Контроль многочисленных условий среды, например, угла наклона метки, расположения метки относительно камеры, условий освещения и т.д., при отсутствии автоматизации на практике представляет собой

сложную задачу и не позволяет нейтрализовать различные погрешности.

- Ограничения в выборе аппаратного обеспечения. У любого оборудования наличие шумов неизбежно и часто оно не отвечает требованиям экспериментов, например, по разрешению камеры, искажениям оптики, чувствительности оптического сенсора и т.д.

Чтобы устранить эти недостатки, эксперименты были перенесены в виртуальную среду Gazebo / ROS. Проведение виртуальных экспериментов обладает следующими преимуществами:

- Проведение экспериментов автономно и требует вмешательства человека только на этапе запуска экспериментов и анализа полученных в ходе экспериментов данных, который проводится по окончанию всех экспериментов.
- Отсутствие ограничений на используемое виртуальное оборудование. Виртуальное оборудование может иметь любые желаемые параметры.
- Виртуальные эксперименты могут быть легко повторены и их результаты могут перепроверены другими заинтересованными исследователями независимо от оригинальных экспериментов.

Данная статья представляет среду для виртуальных экспериментов и пилотное сравнение систем ArUco [Romero-Ramirez et al., 2018] и AprilTag [Olson, 2011] по критерию устойчивости метки к вращениям. Во второй главе представлены результаты предыдущих научных исследований. Глава 3 коротко описывает программные инструменты ROS/Gazebo, использованные для создания виртуальной среды для экспериментов. Глава 4 посвящена организации виртуальных экспериментов; в главе 5 показаны результаты проведенных сравнений, а глава 6 представляет наши заключения.

2 Среда ROS/Gazebo

Робототехническая операционная система (англ. Robot Operating System, ROS) — это быстро растущий фреймворк для робототехнической разработки. Ее архитектура состоит из так называемых узлов (англ. nodes) и потоков данных (англ. topics) для коммуникации между ними. Такая распределенная структура позволяет создавать различные схемы потоков информации, реализовать анализ данных датчиков и контроль движений узлов робота. ROS распространяется в форме минимально функциональных единиц — пакетов (англ. packages). Gazebo представляет собой трёхмерный симулятор, интегрированный в ROS. Он может быть использован как инструмент для визуализации симуляций и имитации характеристик

реального мира, включая освещение, обработку столкновений, гравитацию и другие внешние факторы.

Ряд СКМ обладает своими собственными алгоритмами распознавания в форме ROS пакетов, включая AprilTag, ArUco, Alvar [Woodward et al., 2002] и ChiliTag [Bonnard et al., 2013]. В действительности, ArUco – это универсальная библиотека распознавания, которая может быть использована для распознавания не только ArUco, но и AprilTag, ARTag [Hirzer et al., 2008] [Fiala, 2005a] и ARToolKitPlus [Wagner et al., 2007].

3 Организация эксперимента

Виртуальные эксперименты позволяют исключить влияние большинства факторов внешней среды, которые обычно неустранимы в реальных экспериментах, включая условия освещения, погрешности измерений и шумов сенсоров. Эксперименты, представленные в этой статье, посвящены сравнению двух СКМ в идеальной среде, когда внешние факторы среды не мешают работе алгоритмов распознавания. Результаты таких экспериментов необязательно могут быть повторены в лабораторных или полевых условиях. Однако, они дают понимание поведения сравниваемых СКМ при вращении меток относительно камеры

Таблица 1. Неизменные характеристики виртуальных экспериментов

• **Характеристика**	**Значение**
Разрешение камеры	640 x 480 пикс.
Уровень искажения камеры	0 (идеальная оптика)
Уровень шума камеры	0 (идеальное устройство)
Расстояние	2,0 м.
Диапазон вращения (ось X)	[-180˚; +180˚)
Диапазон вращения (ось Z)	[-90˚; +90˚]
Размер метки	0,4 м. x 0,4 м.
Угол падения света	45˚
Световой спектр	Белый свет
Условия освещения	Равномерное по всей площади метки

Мы создали двух роботов в виртуальной среде: робот-исполнитель, на котором крепится метка, и робот-наблюдатель с камерой, подобный роботу R2D2. Робот-исполнитель спроектирован для модификации положения метки между отдельными распознаваниями: он вращает метку на заранее заданную величину угла в определяемых пользователем пределах (схема вращения показана на Рис. 2). Робот-наблюдатель имитирует статичный

штатив с камерой (Рис. 3). Значения многих параметров эксперимента поддерживаются постоянными во всех экспериментах (см. Таблицу 1) для исключения их влияния на результаты распознавания.

Многочисленные параметры виртуального эксперимента можно контролировать с помощью графического интерфейса, который интегрирован в пакет. Эти параметры включают расстояние между камерой и маркером, уровень шума камеры, диапазоны вращения и количество потоков симуляции. Значения параметров остаются постоянными во всех наших экспериментах, включая условия освещения и отражения, что гарантирует чистоту и объективность эксперимента. Важно подчеркнуть, что виртуальность экспериментов полностью исключает влияние случайных факторов и обеспечивает детерминированность экспериментов.

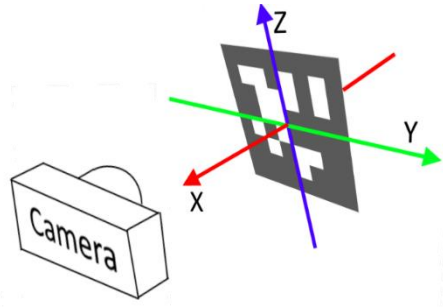


Рис. 2. Схема вращения метки в виртуальном эксперименте.

Эксперимент с каждой меткой проходит по следующему алгоритму:

1. Роботы устанавливаются на постоянном расстоянии друг от друга. Первоначально угол вращения маркера вокруг выбранной пользователем оси равен нулю.
2. Система протоколирования событий – то есть процесса записи в журнал информации о происходящих с координатной меткой событиях – ожидает полсекунды для обнаружения маркера.
3. Система протоколирования событий регистрирует угол наклона и результат обнаружения, который может быть успешным или неудачным.
4. Робот-исполнитель поворачивает маркер на 1 градус вокруг определенной пользователем оси (ось X или Z).
5. Если предел вращения, определяемый пользователем, не достигнут, алгоритм переходит к шагу №2; иначе эксперименты с данной меткой заканчиваются, происходит смена метки и переход на шаг №1.

Система регистрации регистрирует все результаты дважды: первый вывод поступает на консоль, второй - в файл с уникальным именем, включающий в себя семейство тегов (например, ArUco), тип тега (например, 25h7), идентификатор тега, расстояние до камеры, уровень шумов виртуальной камеры и дату эксперимента.

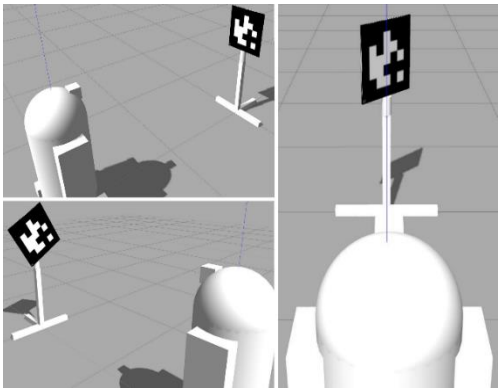


Рис. 3. Дизайн виртуальных экспериментов: начальное положение метки (вверху слева); поворот на 30 градусов вокруг оси X (внизу слева); поворот на 45 градусов вокруг оси Z (справа).

4 Результаты экспериментов

Все маркеры выбранного типа тестируются для сбора статистически значимого объема данных: 100 маркеров ArUco (тип 25h7) и 242 маркера AprilTag (тип 25h7). Маркеры этого типа имеют 25 пикселей кодирования, а расстояние Хэмминга между любыми из них равно или больше 7 (рис. 4). Показатель расстояния Хэмминга оценивает, насколько любые два маркера одного типа отличаются друг от друга. В практическом смысле это позволяет правильно распознавать метку даже в случаях неполных или поврежденных данных с камеры. Одинаковые типы маркеров устраняют разницу между свойствами кодирования СКМ и объемом закодированных данных. Эксперименты проводились с каждым отдельным маркером дважды, чтобы собрать надежные данные о каждом угле обнаружения. Предел вращения маркеров был установлен [-90;90] для оси Z и [-180; 180) для оси X. Таким образом, суммарно было проведено (100 + 242) * (360 + 181) * 2 = 370044 экспериментов. Результаты экспериментов представлены в Таблице 2.

Таблица 2. Усредненные значения успешности распознавания по всем меткам относительно типа и способа их вращения

• **Семейство и тип меток**	**Ось вращения**	
	X	**Z**
• AprilTag – 25h7	• 99,94 %	• 69,96 %
• ArUco – 25h7	• 99,97 %	• 86,07 %

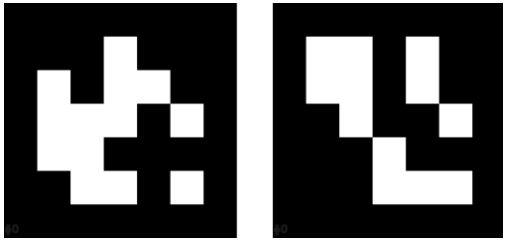


Рис. 4. Примеры меток из эксперимента: AprilTag 25h7 (слева) и ArUco 25h7 (справа).

Результаты экспериментов позволяют сделать вывод, что маркеры семейств AprilTag и ArUco практически нечувствительны к поворотам относительно оси X; однако маркеры семейства ArUco обладают значительно лучшей устойчивостью к поворотам относительно оси Z. Распределения неуспешных обнаружений для каждого семейства маркеров представлены на рис. 5 и рис. 6.

Графики результатов распознавания при вращении относительно оси Z показывают довольно предсказуемое поведение СКМ: увеличение угла поворота уменьшает площадь маркера, видимую для камеры, что приводит к неудачным обнаружениям. Обе СКМ показали наивысшую устойчивость к вращению относительно оси X: для всех углов поворота была зафиксирована почти идеальная частота обнаружения.

5 Заключение

В данной статье представлен ряд экспериментов с маркерными системами AprilTag и ArUco в виртуальной среде и сравнение их устойчивости к вращениям. Обе СКМ показали себя практически инвариантными к вращению меток относительно оси X. Максимальные углы инвариантности распознавания меток ArUco оказались больше, чем углы инвариантности распознавания меток AprilTag: 75° у СКМ ArUco 25h7 против 60° у СКМ AprilTag. Созданная виртуальная среда может быть использована в качестве основы для дальнейших сравнительных исследований, поскольку добавление новой СКМ в проект упрощено. Наша будущая работа сосредоточена на сравнении различных СКМ для их устойчивости к перекрытиям (как систематическим, так и произвольным) и

изучению зависимости максимального расстояния обнаружения от разрешения камеры.

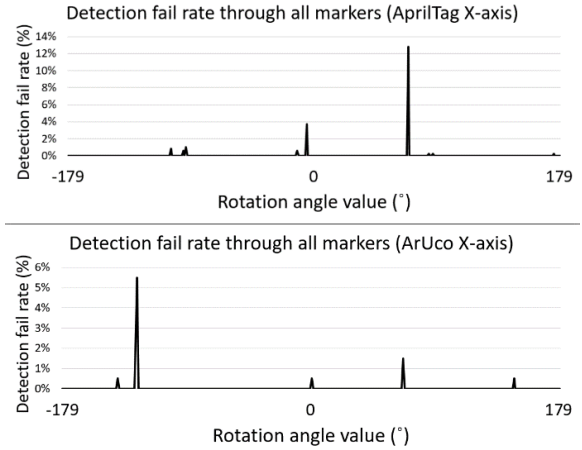


Рис. 5. Результаты экспериментов по вращению относительно оси X: распределение неудачных распознаваний среди всех меток для меток AprilTag (вверху) и ArUco (внизу).

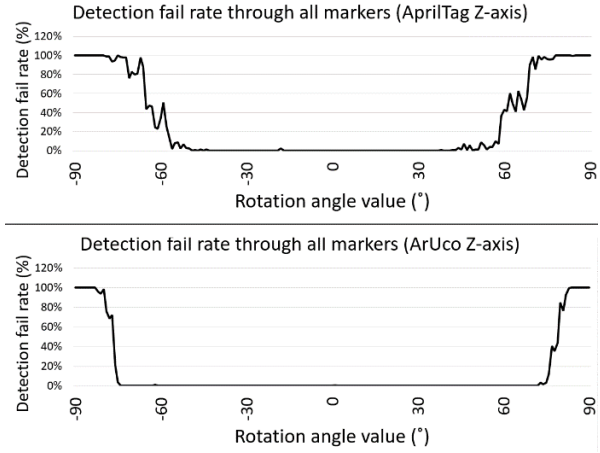


Рис. 6. Результаты экспериментов по вращению относительно оси Z: распределение неудачных распознаваний среди всех меток для меток AprilTag (вверху) и ArUco (внизу).

Список литературы

[Atcheson et al., 2010] Atcheson B., Heide F., Heidrich W. CALTag: High Precision Fiducial Markers for Camera Calibration //VMV. – 2010. – Т. 10. – С. 41-48.

[Bonnard et al., 2013] Bonnard Q. et al. Chilitags 2: Robust fiducial markers for augmented reality and robotics //CHILI, EPFL, Switzerland. – 2013.

[Dhiman et al., 2013] Dhiman V., Ryde J., Corso J. J. Mutual localization: Two camera relative 6-dof pose estimation from reciprocal fiducial observation // IEEE/RSJ Int. Conf. on Intelligent Robots and Systems. – IEEE, 2013. – С. 1347-1354.

[Fiala, 2005a] Fiala M. ARTag, a fiducial marker system using digital techniques //2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. – IEEE, 2005. – Т. 2. – С. 590-596.

[Fiala, 2005b] Fiala M. Comparing ARTag and ARToolkit Plus fiducial marker systems //IEEE Int. Workshop on Haptic Audio Visual Environments and their Applications. – IEEE, 2005. – С. 6.

[Garrido-Jurado et al., 2014] Garrido-Jurado S. et al. Automatic generation and detection of highly reliable fiducial markers under occlusion //Pattern Recognition. – Т. 47. – №. 6. – С. 2280-2292.

[Hirzer et al., 2008] Hirzer M. Marker detection for augmented reality applications //Seminar/Project Image Analysis Graz. – 2008. – С. 1-2.

[Kuriya et al., 2015] Kuriya R., Tsujimura T., Izumi K. Augmented reality robot navigation using infrared marker //24th IEEE Int. Symposium on Robot and Human Interactive Communication. – IEEE, 2015. – С. 450-455.

[Olson, 2011] Olson E. AprilTag: A robust and flexible visual fiducial system //2011 IEEE Int. Conf. on Robotics and Automation. – IEEE, 2011. – С. 3400-3407.

[Romero-Ramirez et al., 2018] Romero-Ramirez F. J., Muñoz-Salinas R., Medina-Carnicer R. Speeded up detection of squared fiducial markers //Image and Vision Computing. – 2018. – Т. 76. – С. 38-47.

[Sagitov et al., 2017] Sagitov A. et al. Comparing fiducial marker systems in the presence of occlusion //Int. Conf. on Mechanical, System and Control Engineering. – IEEE, 2017. – С. 377-382.

[Shabalina et al., 2017] Shabalina K. et al. ARTag, AprilTag and CALTag Fiducial Systems Comparison in a Presence of Partial Rotation: Manual and Automated Approaches //Lecture Notes in Electrical Engineering.

[Wagner et al., 2007] Wagner D., Schmalstieg D. Artoolkitplus for pose tracking on mobile devices. – na, 2007. – С. 139-146.

[Wang et al., 2016] Wang J., Olson E. AprilTag 2: Efficient and robust fiducial detection //IEEE/RSJ Int. Conf. on Intelligent Robots and Systems. – IEEE, 2016. – С. 4193-4198.

[Woodward et al., 2002] Woodward C. et al. Wireless 3d cad viewing on a pda device //Proceedings of the 2nd Asian Int. Mobile Computing Conf. – 2002. – Т. 14. – С. 17.

УДК 004.896

МОДЕЛИРОВАНИЕ МОБИЛЬНОГО РОБОТА АВРОРА ЮНИОР В СРЕДЕ ROS/GAZEBO

К.С. Шабалина (*ks.shabalina@it.kfu.ru*), А.Г. Сагитов (*sagitov@it.kfu.ru*), Е.А. Магид (*magid@it.kfu.ru*)
Высшая школа информационных технологий и интеллектуальных систем, Казанский Федеральный Университет, Казань

Аннотация. Эксперименты являются важным инструментом для разработки надежных алгоритмов управления робототехническими системами (РТС), однако их проведение, как правило, дорогостоящее и требует большого количества времени. До проведения сложных натурных экспериментов рекомендуется осуществить предварительное моделирование и тестирование РТС в симуляционной среде, что существенно удешевляет разработку и повышает безопасность РТС, экспериментаторов и окружающей среды. Обязательным условием моделирования экспериментов является создание достаточно точной модели РТС, которая сохраняет основные физические свойства реального робота. В данной работе представлен процесс разработки модели российского мобильного робота Аврора «Юниор» в среде ROS/Gazebo с целью дальнейшей апробации алгоритмов автономной навигации[1].

Ключевые слова: неголономный робот, симуляция, Аврора Юниор, Gazebo, ROS.

Введение

В последние десятилетия широкое использование симуляций стало неотъемлемой частью исследований в области робототехники. Использование симуляторов помогает удешевить и ускорить разработку РТС, является более безопасным инструментом первичной верификации идей и алгоритмов, а также позволяет тестировать новые концепции и алгоритмы, независимо от наличия требуемого оборудования у научной группы. Моделирование может применяться для всех типов РТС и задач на начальных этапах развития проекта.

[1] Работа выполнена при финансовой поддержке РФФИ (проект № 19-58-70002).

Симуляторы стали прогрессивными инструментами, которые позволяют воспроизводить сложные среды и использовать определяемую пользователем физику, что дает возможность создавать модели РТС, которые ведут себя максимально близко к реальному поведению робота. Gazebo является одним из самых популярных 3D-симуляторов роботов, который успешно используется для симуляции БЛА [Sagitov et al., 2017] и БНР [Afanasyev et al., 2015], [Sokolov et al., 2017], и для проведения различных экспериментов, включая тестирование базовых движений роботов, планирование пути и коллаборативное взаимодействие с другими роботами [Yao et al., 2017], эксперименты с манипуляторами [Qian et al., 2014] и моделирование сценариев поисково-спасательных операций [Kohlbrecher et al., 2014]. Другие специализированные симуляторы (например, UWSim [Kermorgant, 2014]) позволяют моделировать водную среду для автономных подводных роботов.

В рамках задачи по созданию стека навигации и разработке алгоритмов передвижения (англ. locomotion algorithms) и автономной навигации для мобильного робота Аврора «Юниор» была построена модель в симуляторе Gazebo и проведено тестирование алгоритмов управления. В данной работе представлен процесс разработки модели РТС Аврора «Юниор».

2 Робот Аврора «Юниор»

Аврора «Юниор» - мобильный неголономный робот (англ. car-like mobile robot) российского производства компании Аврора Роботикс (см. Рис. 1) [Avrora Robotics, 2018]. Первоначально робот был спроектирован для обучения студентов алгоритмам автономного движения. Для этого «Юниор» выполнен как безопасное решение (вес робота составляет ~43,5 кг) для тестирования и проверки автономного движения как внутри помещения, так и в полевых условиях, а благодаря своим небольшим линейным размерам, он легко транспортируется внутри лабораторных помещений одним или двумя студентами.

Робот обладает рулевым механизмом, соответствующим реальным полноразмерным машинам. Таблицы 1 и 2 демонстрируют линейные размеры и весовые характеристики робота, соответственно. «Юниор» состоит из внешнего корпуса (англ. robot shell), шасси (металлические опоры), источника питания (свинцово-кислотный аккумулятор Delta DTM 1233 L), бортовых датчиков и вычислительного блока (ноутбук модели Acer TravelMate P2). Другим немаловажным преимуществом робота является открытое программное обеспечение (ПО) и возможность усовершенствования аппаратного обеспечения. Последнее может быть легко заменено или могут быть добавлены новые элементы в систему РТС (например, дополнительные датчики). «Юниор» имеет предустановленную

ОС Linux Ubuntu 16.04 LTS и Робототехническую Операционную Систему ROS версии Kinetic. Компания-производитель также предоставляет открытые пакеты контроля и управления натурным роботом.

2.1 Робототехническая операционная система

Робототехническая Операционная Система (англ. Robot Operating System, ROS) представляет собой популярный открытый фреймворк для командной разработки программного обеспечения роботов. Атомарной единицей среды является пакет (англ. package), который должен решать определенную задачу для конкретной модели робота. Согласно идеологии, ROS пакеты могут быть улучшены любым разработчиком, а новые пакеты – добавлены в общую базу знаний с описанием инструкции по использованию и составных модулей ROS.

Для моделирования робота Аврора «Юниор» был выбран симулятор Gazebo: симулятор является открытым ПО, имеет прямую интеграцию с ROS, что позволяет легко разрабатывать программные пакеты для робота и его модели, а также легко тестировать разработанные алгоритмы в режиме симуляции. Gazebo предоставляет достаточно точную симуляцию различных типов существующих роботов (БНР, БЛА, манипуляторы), а также позволяет создавать собственные модели роботов. Симулятор учитывает заданную физику мира и модели, что позволяет проводить эксперименты (например, верификацию новых алгоритмов) с роботами в режиме симуляции. Также симулятор позволяет создавать желаемую среду для робота, добавлять различные препятствия и другие объекты, и предоставляет различные плагины, которые симулируют различные типы датчиков. Формат описания модели в Gazebo представлен в двух способах – SDF (Simulation Description Format) и URDF (Unified Robot Description Format). Симулятор поддерживает STL и DAE форматы полигональных трехмерных сеток - *мешей* (англ. mesh, структурная сборка из вершин, ребер, граней 3D модели, состоящая из многоугольников) для объектов.

Для описания модели робота Аврора «Юниор» в Gazebo был использован формат URDF (рус. Унифицированный Формат Описания Робота. URDF представляет собой файл формата XML, который описывает робота с помощью определенных тегов. Существуют основные теги, которые необходимы для описания визуальной структуры робота (его «скелета»): link и joint, звено и сустав (сочленение), соответственно. Звенья соединяются между собой с помощью суставов, представляя собой древовидную структуру модели робота. Другие теги предназначены для описания внутренних характеристик звеньев и суставов, добавления плагинов датчиков и их настройки, указания контроллеров суставов и их характеристик.

Рис. 2. Аврора «Юниор» робот.

3 Модель робота в Gazebo

Проект модели робомобиля состоит из нескольких пакетов, разделенных по функциональному предназначению: *avrora_description, avrora_gazebo, avrora_control*. Такое разделение представляет удобный и понятный способ к понимаю основных задействованных модулей модели. Пакет avrora_description содержит описание модели для Gazebo: описание модели в URDF, используемые трехмерные сетки и материалы модели. Пакет avrora_gazebo включает конфигурацию модели для её правильной загрузки в симуляторе Gazebo. Пакет avrora_control описывает конфигурацию суставов модели и их геометрию: тип сустава, коэффициенты ПИД регулятора и другие ROS настройки контроллеров.

Основными элементами робомобиля, представляющими модель, были выбраны следующие: base_link (внешний каркас робота), right_steer_link (правая часть рулевой рейки, часть будущего механизма системы подвески робота согласно системе Аккермана), right_steer_wheel (переднее правое колесо), right_drive_wheel (заднее правое колесо), left_steer_link (левая часть рулевой рейки, часть будущего механизма системы подвески робота согласно системе Аккермана), left_steer_wheel (переднее левое колесо), left_drive_wheel (заднее левое колесо). Для того, чтобы модель робомобиля визуально соответствовала реальному роботу, были использованы трехмерные сетки частей робота для данных элементов и некоторых датчиков, рассмотренных ниже (САПР-модель была представлена компанией-производителем) (см. Рис. 2).

Таблица. 1. Линейные размеры робота

• **Параметр**	**Размер (мм)**	**Размер (м)**
Диаметр диска	204	0,204
Диаметр колеса	260	0,26
Ширина колеса	110	0,11
Продольная длина всего робота	1112	1,112
Высота робота	570	0,57
Ширина робота	650	0,65

Таблица. 2. Весовые характеристики элементов Авроры «Юниор»

• **Элемент**	**Вес (кг)**
Колесо	0,9
Корпус робота	8,5
Крышка робота	3,0
Датчик Hokuyo	0,16
Датчик Kinect	0,45
Вес всего робота	43,5

Все звенья модели были заданы в пропорциях, соответствующих пропорциям и размерам реального робота. Чтобы адаптировать инерционные характеристики робота к модели, был рассчитан упрощенный инерционный тензор для её звеньев. В процессе была заменена сложная структура трехмерной сетки каркаса робота (base_link) сплошным кубоидным объектом (англ. solid cuboid); структуры трехмерных сеток рулевых колес (right_steer_wheel, left_steer_wheel) заменены на сплошной цилиндр (англ. solid cylinder) для вычисления приближенных тензоров инерции звеньев модели. Коллизии звеньев были указаны с учетом структуры трехмерных сеток всех звеньев и в том же размере, что и звенья. URDF формат позволяет описывать следующие типы суставов: призматический (англ. prismatic), вращательный с лимитами (англ. revolute), вращательный без лимитов (англ. continuous), «плавающий» (англ. floating) и неподвижный (англ. fixed). Подробные определения суставов представлены ниже. Неподвижные соединения жестко соединяют два звена друг с другом; в конструируемой модели были использовали эти соединения для жесткого прикрепления датчиков робота к основному корпусу.

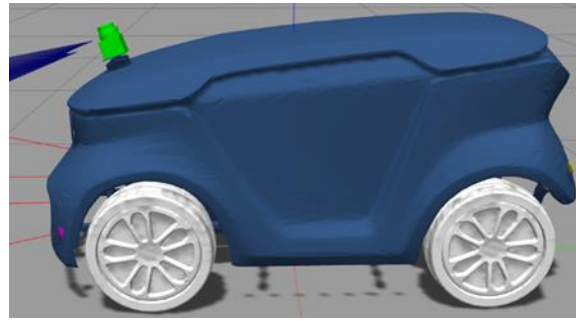
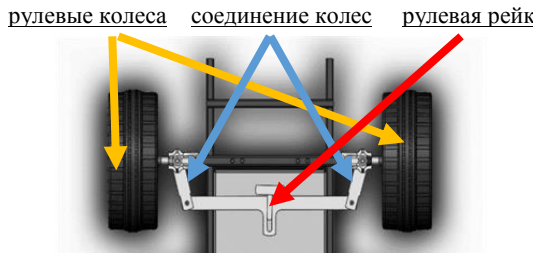


Рис. 3. Модель Авроры «Юниор» в среде ROS/Gazebo

Вращательный сустав без лимитов позволяет вращать шарнир вокруг указанной оси без верхнего и нижнего пределов угла поворота; такой тип был использован для моделирования движения колес робота: *left_steer_wheel_joint, left_drive_wheel_joint, right_steer_wheel_joint, right_drive_wheel_joint*. В свою очередь, вращательный тип сустава с ограничением был применен к рулевым механизмам модели (*left_steer_joint and right_steer_joint*), для последующей разработки системы подвески и управления моделью согласно системе Аккермана (см. Рис. 3).

Рис. 4. Схема шасси робота Авроры «Юниор».



Так как на реальном роботе используются два мотора для движения задних колес (на каждое колесо отводится отдельный мотор, однако они вместе контролируются через единый ROS пакет) и один мотор для организации рулевого механизма, были сконфигурированы следующие ROS-контроллеры для суставов: контроллеры положения *SteerRight_controller* и *SteerLeft_controller* (ROS *JointPositionController*), контроллеры усилий *EffortDriveRight_controller* и *EffortDriveLeft_controller* (ROS *JointEffortController*). Контроллеры положения относятся к суставам рулевого механизма, контроллеры усилий относятся к суставам задних колес (так как на реальном роботе моторами оснащены два задних колеса). Рисунок 4 показывает финальную схему работы контроллеров в среде ROS.

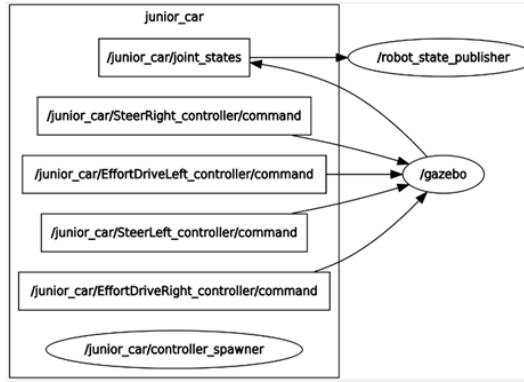


Рис. 5. Визуализация в rqt_graph топиков модели Аврора "Юниор" (топики датчиков не визуализированы).

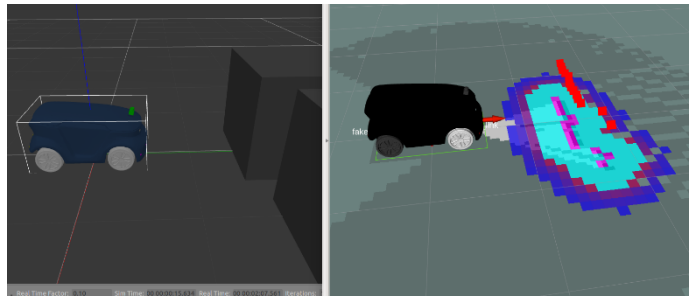


Рис. 6. Слева - модель робота Аврора "Юниор" в Gazebo, справа - визуализация данных с датчиков и топиков в Rviz.

Таблица 3 представляет список датчиков робота Аврора «Юниор» и их имплементация в среде ROS и Gazebo. Модели датчиков были помещены в том же положении и ориентации, как и на реальном роботе, для получения точных данных с датчиков. Все плагины датчиков были сконфигурированы согласно техническому описанию существующих датчиков. Для визуализации датчиков использовалось ПО Rviz (входит в фреймворк ROS). Пример визуализации виртуального датчика лазерного дальномера изображен на рисунке 5.

Таблица. 3. Датчики робота и их плагины в ROS/Gazebo

• Датчик	Пакет в ROS	Плагин в Gazebo
ЛИДАР Hokuyo	hokuyo_node	laser_controller
Microsoft Kinect	freenect_stack	camera_plugin
GPS	ur_nmea_driver	novatel_gps_sim
УЗ датчики	ur_rangefinder_driver	sonar
IMU	myahrs_driver	imu_plagin

4 Заключение

В данной работе было представлено моделирование робота Аврора «Юниор» в симуляторе Gazebo, среде ROS. Для построения модели был использован формат URDF, поддерживаемый симулятором. Основные элементы Авроры «Юниор» были импортированы из трехмерных сеток, предоставленных компанией-производителем робота. Были добавлены соответствующие контроллеры для моделирования рулевого управления модели «Юниор» и соответствующие реальному роботу датчики.

В рамках будущей работы будет выполнена настройка ПИД-регуляторов для контроллеров модели и разработана симуляция рулевого управления и движения согласно модели Аккермана. По завершении этапа моделирования робота, готовая модель будет применена для задач кооперативного поиска пути в моделируемой городской среде [Andreychuk et al., 2017].

Список литературы

[Afanasyev et al., 2015] Afanasyev I., Sagitov A., Magid E. ROS-based SLAM for a Gazebo-simulated mobile robot in image-based 3D model of indoor environment // 2015 International Conference on Advanced Concepts for Intelligent Vision Systems – Springer, Cham 2015. – C. 273-583.

[Andreychuk et al., 2017] Andreychuk A., and Yakovlev K. Applying MAPP Algorithm for Cooperative Path Finding in Urban Environments. // 2017 International Conference on Interactive Collaborative Robotics. – Springer, Cham, 2017. – C. 1-10.

[Avrora Robotics, 2018] Avrora Robotics'18, **-** https://avrora-robotics.com/ru/.

[Kermorgant, 2014] Kermorgant O. A dynamic simulator for underwater vehicle-manipulators // 2014 International Conference on Simulation, Modeling, and Programming for Autonomous Robots – Springer, Cham 2014. – C. 25-36.

[Kohlbrecher et al., 2014] Kohlbrecher, S., Meyer, J., Graber, T., Petersen, K., Von Stryk, O. and Klingauf, U. Kowalsky R. Robocuprescue 2014-robot league team hector Darmstadt (Germany) // RoboCupRescue, 2014.

[Qian et al., 2014] Qian, W., Xia, Z., Xiong, J., Gan, Y., Guo, Y., Weng, S., Deng, H., Hu, Y. and Zhang, J. Manipulation task simulation using ros and gazebo // 2014

International Conference on Robotics and Biomimetics (ROBIO, 2014) – IEEE 2014. – C. 2594-2598.

[Sagitov et al., 2017] Sagitov A., and Gerasimov Y. Towards DJI Phantom 4 Realistic Simulation with Gimbal and RC Controller in ROS/Gazebo Environment // 2017 International Conference on Developments in eSystems Engineering, –IEEE, 2017. – C. 262-266.

[Sokolov et al., 2017] Sokolov, M., Afanasyev, I., Lavrenov, R., Sagitov, A., Sabirova, L. and Magid, E. Modelling a crawler-type UGV for urban search and rescue in Gazebo environment // 2017 International Conference on Artificial Life and Robotics (ICAROB 2017), –2017. – C. 360-362.

[Yao et al., 2017] Yao, W., Dai, W., Xiao, J., Lu, H. and Zheng, Z. A simulation system based on ros and gazebo for robocup middle size league //2017 International Conference on Robotics and Biomimetics (ROBIO, 2017) – IEEE 2017. – C. 54-59.

Материалы круглого стола «РЭП: робототехника, этика, право»

УДК 004.83

ИНТЕЛЛЕКТУАЛЬНЫЕ И АВТОНОМНЫЕ СИСТЕМЫ – ЭТИЧЕСКИЕ АСПЕКТЫ ПРИМЕНЕНИЯ

П.М. Готовцев (*gotovtsevpm@gmail.com*)
НИЦ Курчатовский институт, Москва

Аннотация. Данная работа представляет собой обзор ряда международных инициатив в области этики интеллектуальных и автономных систем (ИИ/АС). Представлена общая постановка вопроса, в том виде, в котором она отражается в большинстве работ, а именно взаимодействие с человеком ИИ/АС в момент принятия этими системами решения и социальные изменения, связанные с внедрением таких систем. Представлена информация об инициативе Института инженеров в области электротехники и электроники (Institute of Electrical and Electronics Engineers – IEEE), направленной на разработку ряда международных стандартов, относящихся к области этики ИИ/АС. Показаны основные задачи, выделенные в рамках данной инициативы IEEE.

Ключевые слова: правила, интеллектуальные и автономные системы, этика технических систем, человеко-машинное взаимодействия, междисциплинарные исследования, технические стандарты.

Введение

Искусственный интеллект – это словосочетание сегодня звучит в совершенно разном контексте. Его используют как для фантастических сверхразумных машин в кинематографе, так и для продвижения «умных» приложений в обычные смартфоны. Хотя, говоря технически корректно некоторые из таких приложений могут попасть по термин интеллектуальные и автономные системы (ИИ/АС) - именно такая терминология используется применительно к многочисленным «умным» системам, которые так активно входят в нашу жизнь в последние годы. Основной особенностью таких систем, является их способность автономно принимать решения, опираясь на имеющиеся данные. Такой системой может быть, например, автопилот автомобиля, который принимает решения как, с какой скоростью и по какому маршруту ехать. То же можно сказать о программах, которые анализируют наше поведение в интернете – активность в социальных сетях, поисковые запросы, местоположение по

комментариям под фотографиями и т.д. Эти программы, обработав данные дают пользователям советы, конечно, право пользователей следовать или не следовать этим советам, но очень часто получается, что советы оказываются весьма полезными. Таким образом, мы сталкиваемся с ситуацией, когда машины принимают решения, которые непосредственно оказывают влияние на нашу жизнь, и по мере дальнейшего развития технологий мы в праве ожидать что областей применения таких «умных» машин будет становиться все больше. Более того, сегодня наиболее ценные и интересные результаты во многих областях от геномных исследований до интернета вещей получаются там, где проводится анализ больших данных. То есть машины могут принимать решения там, где человек уже не может – при анализе больших данных. Итак можно сказать, что сегодня взаимодействие человека и машин обретает новые особенности, Еще раз обратим внимание на то, что речь идет не о сверхразумной искусственном интеллекте который сейчас активно обсуждается [Bostrom 2014] [Bostrom et al 2011], а о достаточно ограниченных по своим возможностям в сравнении с человеком программных продуктах, которые очень хорошо решают одну определенную задачу и не способны одновременно управлять автомобилем и анализировать медицинские данные что бы поставить диагноз пациенту.

1 Вопросы применения ИИ/АС

Раз мы сказали о новых особенностях взаимодействия человека и машины, то необходимо попробовать сформулировать вопросы, которое в результате этого возникают [Broadbent 2017] [Bostrom et al 2011]. Начнем с некоторых ситуаций:

Автономное транспортное средство везет пассажиров. Водителя нет, роль «таксиста» выполняет ИИ/АС. На дороге встретилась авария, есть пострадавшие, а машины скорой помощи еще нет. Что делать в этой ситуации – объехать место аварии, остановится и дать возможность пассажирам убедиться, что скорая в пути? А если среди пассажиров врач, который может помочь?

Домашний робот, осуществляет простейшую уборку пола. Для совершенствования своего обучаемого программного обеспечения передает все данные компании-разработчику. Только вот этот незаметный домашний робот, по сути, передает данные о частной жизни пользователя, организации, которая никак не обязана и не способна заботиться об их сохранности.

Программа, анализирующая поведение пользователя в сети, может получить о пользователе те данные, которые он предпочел бы скрыть.

Кроме того, возникают многочисленные вопросы, связанные с взаимодействием человек – машина, включая выстраивание диалога с программой-советчиком, степень доверия к ИИ/АС, взаимодействия робототехнических систем и человека и т.д. [Grinbaum et al 2017] [Wiener 1960] [Lemagnan et al 2017].

Другим примером являются интеллектуальные программы советчики [Noothigattu et al. 2017] [Gotovtsev et al 2008] [Flores-Loredo et al 2005]. Использование таких программ поднимает целый ряд этических вопросов, среди которых можно отметить следующие:

1. Афилированность программ-советчиков;
2. Влияние таких программ на способность пользователей самостоятельно принимать решение;
3. Может ли, например, программа, анализирующая состояние потенциального заемщика, учесть особые факторы, связанные со сложной жизненной ситуацией?
4. Программа-советчик и опыт оператора – взаимодополнение или противопоставление?

Немаловажным является вопрос анализа метаданных, уже упомянутый выше, а также производимые интеллектуальными системами продукты, как в области искусства [Knight 2011] [Elgammal et al 2017], так и товаров потребления [Yoon et al 2017].

Таким образом, можно сказать, что вопросы этики ИИ/АС возникают во многих областях, где такие системы находят сейчас активное применение. От банковских систем и программ-советчиков, до беспилотных мобильных систем (автомобили и дроны-доставщики) и домашних роботов.

2 Инициатива IEEE

Вопросов, вроде указанных в предыдущем разделе, сегодня возникает все больше, и все они складываются в то направление исследований, которое сегодня получило название этика ИИ/АС [Havens 2016] [EAD 2019]. Исследования в этой области привели к появлению ряда инициатив по исследованию в данной области [Grinbaum et al 2017] [Bostrom et al 2011]. Среди таких инициатив следует выделить глобальную инициативу Института инженеров в области электротехники и электроники (Institute of Electrical and Electronics Engineers – IEEE, https://www.ieee.org/), в рамках которой начата работа по созданию нормативно-технических документов, которые заложили бы основу этичного поведения в разрабатываемые системы с ИИ/АС. Первым шагом в этой работе является создание рекомендаций, направляющих разработчиков ИИ/АС на этически обоснованные решения при разработке и применении систем ИИ/АС [EAD 2019]. Эти рекомендации распространяются как на ситуацию, в которой

система с ИИ/АС или автономная система принимает решение в какой-либо этической ситуации, так и на ситуацию, в которой применение ИИ/АС или автономных систем приводит к каким-либо социальным последствиям в виде сокращения определенного персонала. Таким образом, можно выделить два направления исследований [Havens 2016]:

- непосредственно этику ИИ/АС, как этические проблемы, вызванные работой машин, самостоятельно принимающих решения и так или иначе взаимодействующих с человеком;
- этические проблемы, связанные с массовым внедрением ИИ/АС, которые могут привести к сокращениям персонала, занятого рутинными работами и другим социальным вызовам.

В этой статье мы не будем акцентировать внимание на втором вопросе, так как сегодня еще нет достаточно однозначного мнения о том, как и насколько повлияет на рабочие места эффект от массового внедрения ИИ/АС, как это скажется на различных слоях общества. Будет ли только эффект сокращения занятости или это приведет еще и к росту экономики и появлению, например, безусловного дохода? Кроме того, многие аналитические работы концентрируются на сокращении числа рабочих мест в одних областях, но не учитывают потенциала роста других. Так, например, не учитывается рост биотехнологической индустрии, сельского хозяйства, авиаперевозок, где в ряде развитых стран отмечается даже недостаток рабочей силы. В общем это благодарное поле для исследований ученым из многих областей наук, которое стало актуальным со времен промышленной революции и с начала массового внедрения автоматизации [Wiener 1960] [Markoff 2015].

Очевидно, что в конечном итоге все исследования в этой области ведут к созданию нормативных документов, чем и занимается IEEE (https://ethicsinaction.ieee.org/). При этом, что этическое регулирование ИИ/АС и автономных систем не ведет к ограничениям или запрету в развитии таких технологий. Например, система «умный дом» может вызвать помощь, если определит, что человек в доме, например, потерял сознание. Эта система принимает решение, вызывая помощь, и тем самым, она может спасти жизнь человеку. Ограничивать развитие таких систем, значит лишать шанса многих людей в будущем.

В 2017 году IEEE запустило проект под названием «Глобальная инициатива по этике интеллектуальных и автономных систем». В рамках этой инициативы были собраны мнения более чем 2000 специалистов, как ученых из разных областей наук, так и инженеров. На основе этой информации был разработан концептуальный документ, цель которого поднять вопросы, связанные с использованием ИИ/АС и их взаимодействия с человеком, а также указать направления для исследований в этой области [EAD 2019]. Охватываемые документов вопросы включают в себя:

- Обсуждение возможности создания методологии для разработчиков ИИ/АС, учитывающей этичность поведения таких систем в будущем;
- Вопросы связанные с валидацией ИИ/АС в части их взаимодействии с человеком;
- Вопросы связанные с базовыми подходами к разработке ИИ/АС с учетом этически обусловленного взаимодействия с человеком. Причем в данном случае авторы делают акцент на особенности устоявшихся моральных норм в различных сообществах и различные виды ценностей, не ограничиваясь только базовыми ценностями западной цивилизации, но и учитывая особенности восточных и африканских мировоззрений;
- Вопросы подготовки специалистов, связанные с этикой ИИ/АС;
- Юридические вопросы, в частности юридического статуса ИИ/АС, ответственности разработчика и ответственности пользователя;
- Потенциальные вызовы применения ИИ/АС в различных областях.

Опираясь на данный документ [EAD 2019], IEEE создает целую группу стандартов, посвященных этическим аспектам ИИ/АС. На сегодня эта группа включает в себя 13 различных стандартов, получивших проектные номера, начиная с P7000. Они охватывают ряд аспектов связанных с применением ИИ/АС при обработке персональных данных, в том числе различных социальных и возрастных групп, взаимодействию машин и людей (роботы-сиделки, распознавание лиц и т.д.), фильтрации фейковых новостей в информационном потоке, онтологиям для ИИ/АС и самое важное вопросам валидации ИИ/АС и этическим аспектам их разработки (https://ethicsinaction.ieee.org/). Следует отметить, что данные стандарты являются нормативно техническими документами, то есть должны стандартизировать то, как проводить определенные работы по разработке системы или то каким требованиям должна соответствовать в ходе своей эксплуатации ИИ/АС. Таким образом, возможно впервые разработкой технических стандартов занимается междисциплинарная группа ученых из самых разных областей знаний.

3 Исторические предпосылки инициативы IEEE

Вопрос этики ИИ/АС возник не сегодня, Норберт Винер затрагивает вопросы этики ИИ в своей статье [Wiener 1960] и в дополнительных главах второго издания своей «Кибернетики» [Wiener 1965]. Основная мысль, которую он постулирует в этих работах, заключается в том, что машину могут быть опасны для человека и непредсказуемы. При этом он отмечает, что даже понимая в деталях как работает машина, оператор может не успеть понять, что ее рассуждения идут к негативному сценарию или даже не

успеть понять, что машина уже «приняла решение» и работает над осуществлением этого сценария. Уже к моменту написания статьи [Wiener 1960] разница в скорости обработки информации требующей вычислений между человеком и современными на тот момент вычислительными машинами была значительной. Именно эта разница, помноженная на неполную конкретизацию человеком своих желаний и может привести по мнению автора к негативным последствиям. Как видно из вышесказанного, поставленные Норбертом Винером вопросы и сегодня остаются актуальными, при этом зачастую сегодня оператор и разработчики не имеют информации о том, что происходит в данный момент времени в созданных ими интеллектуальных агентах. Уже классическим примером являются масштабные искусственные нейронные сети (ИНС) глубокого обучения, прописать детально работу некоторых из них после обучения не в состоянии даже сами разработчики. Тем не менее этот крайне эффективный метод сегодня получает широчайшее распространении при анализе данных, включая поиск метаданных, а также разработаны алгоритмы, позволяющее таким ИНС иметь «ячейки памяти» для сохранения промежуточных состояний [Graves et al 2016] [Kirkpatrick et al 2017]. В то время как для взаимодействия ИНС с людьми при решении совместных задач уже приходится разрабатывать новые алгоритмы [Crandall et al. 2017].

4 Заключение

Тем не менее, многие читающие сейчас эти строки разработчики ИИ/АС наверняка думают что-то вроде – ну вот сейчас придут гуманитарии и начнут нас своей этикой ограничивать. Во-первых, такие исследования удел не только философов и культурологов, но и инженерного сообщества. Так как перед разработчиками ИИ/АС возникает задача создать алгоритмы способные в области своей деятельности вести себя с учетом тех этических основ, которые будут выработаны научным сообществом. Решение этих задач может стать еще одним из двигателей прогресса в области ИИ/АС, помогая построить мост между формальными математическими системами и гуманитарными понятиями как этика, нравственность и мораль [Карпов и др 2018].

В завершение следует отметить, что решение задач в области этики ИИ, очевидно, требует междисциплинарного подхода. Этот подход должен выражаться не только в исследованиях с участием ученых из разных областей науки, но и появлением в учебных курсах специалистов в области ИИ вопросов этики, и наоборот, предметов, рассказывающих о современном уровне технологий для тех студентов-гуманитариев, кто интересуется технической этикой в целом.

Список литературы

[Карпов и др 2018] В.Э. Карпов, П.М. Готовцев, Г.В. Ройзензон. К вопросу об этике и системах искусственного интеллекта // Философия и общество. 2018, №2(87) С.84-105

[Bostrom 2014] N. Bostrom, Superintelligence: Paths, Dangers, Strategies. 2014.

[Bostrom et al 2011] N. Bostrom and E. Yudkowsky, The Ethics of Artificial Intelligence // Cambridge Handb. Artif. Intell., pp. 1–20, 2011.

[Broadbent 2017] E. Broadbent, Interactions With Robots: The Truths We Reveal About Ourselves // Annu. Rev. Psychol., vol. 68, no. 1, pp. 627–652, Jan. 2017.

[Crandall et al. 2017] J. W. Crandall et al., "Cooperating with Machines," Mar. 2017.

[EAD 2019] Ethically aligned design: A vision for prioritizing human wellbeing with artificial intelligence and autonomous systems. 2019.

[Elgammal et al 2017] A. Elgammal, B. Liu, M. Elhoseiny, and M. Mazzone, "CAN: Creative Adversarial Networks, Generating 'Art' by Learning About Styles and Deviating from Style Norms," 2017.

[Flores-Loredo et al 2005] Z. Flores-Loredo, P. H. Ibarguengoytia, and E. F. Morales, On Line Diagnosis of Gas Turbines using Probabilistic and Qualitative Reasoning // Proceedings of the 13th International Conference on, Intelligent Systems Application to Power Systems, 2005, vol. 2005, pp. 297–301.

[Gotovtsev et al 2008] P. M. Gotovtsev, V. N. Voronov, and D. S. Smetanin, Analysis of the coolant condition with the help of artificial neural networks, // Therm. Eng., vol. 55, no. 7, pp. 552–557, Jul. 2008.

[Grinbaum et al 2017] B. A. Grinbaum, R. Chatila, L. Devillers, J. Ganascia, C. Tessier, and M. Dauchet, Ethics in Robotics Research // IEEE Robot. Autom. Mag., vol. 24, no. 3, pp. 139–145, 2017.

[Graves et al 2016] A. Graves et al., Hybrid computing using a neural network with dynamic external memory // Nature, 2016.

[Havens 2016] J. Havens, Heartificial Intelligence: Embracing Our Humanity to Maximize Machines. 2016.

[Kirkpatrick et al 2017] J. Kirkpatrick et al., Overcoming catastrophic forgetting in neural networks // Proc. Natl. Acad. Sci. U. S. A., p. 201611835, Mar. 2017.

[Knight 2011] H. Knight, Eight Lessons learned about Non-verbal Interactions through Robot Theater // ICSR: International Conference on Social Robotics, 2011, pp. 42–51.

[Lemagnan 2017] S. Lemaignan, M. Warnier, E. A. Sisbot, A. Clodic, and R. Alami, Artificial cognition for social human–robot interaction: An implementation // Artif. Intell., vol. 247, pp. 45–69, Jun. 2017.

[Markoff 2015] J. Markoff, Machines of Loving Grace. Ecco Pr, 2015.

[Noothigattu et al. 2017] R. Noothigattu et al., "A Voting-Based System for Ethical Decision Making," 2017.

[Wiener 1960] N. Wiener, Some Moral and Technical Consequences of Automation // Science (80-.)., vol. 131, no. 3410, pp. 1355–1358, 1960.

[Wiener 1965] N. Wiener, Cybernetics : or, Control and communication in the animal and the machine. M.I.T. Press, 1965.

[Yoon et al 2017] H.-S. Yoon et al., CAD/CAM for scalable nanomanufacturing: A network-based system for hybrid 3D printing // Microsystems Nanoeng., vol. 3, p. 17072, Sep. 2017.

УДК 004.8, 17, 519.7

ОТ ПОДРАЖАТЕЛЬНОГО ПОВЕДЕНИЯ К ЭМПАТИИ В СОЦИУМЕ РОБОТОВ

В.Э. Карпов (karpov-ve@yandex.ru)
Национальный исследовательский центр "Курчатовский институт", Москва
МФТИ

Аннотация. В работе рассматриваются некоторые конструктивные аспекты явного введения в систему управления роботом понятия субъективного Я. Показано, что эта компонента позволяет естественным образом реализовать такие феномены, как подражательное поведение, социальное обучение и эмпатия. В основе разработанных моделей лежит понятие степени близости наблюдаемого контрагента к субъекту. Делается предположение, что эти модели могут стать основой для создания такого механизма адаптивного поведения, который называется моралью.[1]

Ключевые слова: групповая робототехника; социальные сообщества роботов; подражательное поведение; эмпатия; мораль; субъективное Я.

Введение

Одним из направлений групповой робототехники являются т.н. модели социального поведения. Ключевой задачей этого направления является создание конструктивных моделей, позволяющих реализовывать феномены социальной организации в группах искусственных агентов [Карпов, Карпова, Кулинич, 2019]. Основная идея парадигмы моделей социального поведения (МСП) заключается в том, чтобы рассматривать принципы организации сообществ роботов с точки зрения некоторого универсального адаптационного механизма. Сам вопрос о видах поведения, в т.ч. – социального – является до сих пор открытым. Так, с одной стороны психологии и биологи говорят о четырех основных формах поведения: пищевое, оборонительное, половое и исследовательское, см., например, [Лурия, 2007]. С другой стороны, те же этологи или специалисты в области социальной биологии выделяют дополнительные, специфические формы и

[1] Работа выполнена при частичной поддержке грантов РФФИ 17-29-07083 офи_м (семиотические аспекты) и 16-29-04412 офи_м (эмпатия, модели поведения).

модели поведения, необходимые для организации социума, такие, как контагиозное (заразное), подражательное, агрессивное и пр. [Зорина, Полетаева, Резникова, 2002]. Многие из этих механизмов достаточно сильно пересекаются друг с другом, зачастую представляя собой лишь некоторое условное обозначение проявления тех или иных базовых механизмов. Так, возникает сомнение в существовании агрессивного поведения как самостоятельной, специфичной сущности (см. [Карпова, Карпов, 2018]), хотя о самой агрессии написано множество фундаментальных трудов (правда, чаще представляющих объемное описания тех или иных феноменов и проявлений того, что мы называем агрессией, см. [Лоренц, 1994], [Ениколопов, Кузнецова, Чудова, 2014]) и др.

Важным вопросом парадигмы МСП является определение соответствия уровня развития архитектуры искусственного агента той или иной форме или механизму социального поведения. Такие механизмы, как когезия (стремление держаться вместе), обучение (в простейших формах), доминирование, дифференциация функций и некоторые другие вполне реализуемы на рефлекторном уровне организации системы управления. Напротив, механизмы социального обучения и подражательное поведение – это уже прерогатива когнитивной архитектуры. Здесь мы не будем касаться вопросов организации когнитивного уровня агентов. Эти вопросы относятся к исследованиям в области BDI-архитектур (см., например, [Кулинич, 2018]) и в определенном смысле – прикладной семиотики [Осипов и др., 2018]. Мы затронем в этой работе лишь один аспект когнитивных архитектур – элемент, называемый "субъективное Я". Далее будет показано, как введение этого элемента позволяет реализовать схему подражательного поведения, а также реализовать модель социального обучения. Кроме того, наличия этого "субъективного Я" позволяет подойти к решению вопросов эмпатии и даже некоторых аспектов этического поведения.

1. Субъективное Я

В социальной психологии субъективное Я (далее – С.Я.) – это то, что человек сам о себе думает и знает. С.Я. называют также Я-познающее. Более строго, в психологии С.Я. состоит из знаний трех типов, которые вытекают из активной деятельностной роли личности и включают такие компоненты, как: (1) континуальность, т.е. осознание себя одним и тем же человеком в течение онтогенеза; (2) знания об индивидуальности, о своем отличии от других; (3) воля, чувство личного контроля [Знаков, 2005]. В прикладной семиотике существует аналогичное понятие – субъект деятельности. Включение этого понятия в модель мира формирует т.н.

картину мира [Осипов и др., 2018]. При этом полагается, что одной из основных функций субъекта деятельности является целеполагание.

Следует отметить, что чаще всего, вне рамок семиотического подхода, С.Я. определяется неявным образом. В любом случае оценка близости текущего состояния к целевому, определение местоположения и пр. – все это происходит относительно самого агента. Явное же задание компоненты С.Я. не только позволяет унифицировать описание картины мира, но, прежде всего, дает возможность создавать модели различных форм поведения, описывающих взаимодействие между агентами.

Прежде, чем будут рассмотрены конструктивные аспекты применения С.Я. для описания некоторых форм поведения, сделаем важное замечание. Далеко не все модели социального поведения требуют введения компоненты С.Я., несмотря даже на внешнюю сложность и кажущуюся необходимость наличия семиотической модели. Таким примером является т.н. контагиозное ("заразное") поведение.

Контагиозное поведение. Суть этого поведения достаточно проста: в отсутствии явного стимула, регистрируемого сенсорной системой индивида, за счет внешнего сигнала могут быть запущены его различные поведенческие реакции. Например, сигнал опасности подхватывается остальными членами группы, заставляя стаю птиц сниматься с места, даже если птицы не видят непосредственной угрозы. Это – простейший тип сотрудничества, реализация принципа "делай, как я" [Тинберген, 1993]. Важно, что это не подражание, так как реагирующие особи не научаются выполнять определенные движения, наблюдая за действиями других. В [Карпов, Карпова, Кулинич, 2019] показано, как реализуется контагиозное поведение без привлечения С.Я., а также понятия "Иной" ("Ты").

Подражательное поведение. Реализация этого феномена требует ответа на следующие вопросы: (1) чему следует подражать, (2) почему это следует делать и (3) кому надо подражать. Будем называть агента, реализующего подражательное поведение, наблюдателем, а того, чьему поведению он подражает – контрагентом (в этологии – конспецификом).

Первый вопрос – это определение того, чему, собственно надо подражать. Т.е. необходимо понять, что делает контрагент или в каком состоянии он находится. Разумеется, определение или распознавание состояния агента – это крайне сложная задача, связанная с анализом сцен, причем зачастую – динамических. Однако здесь зачастую используется один чисто технический "трюк". Контрагент сам сигнализирует о том, в каком состоянии он находится в текущий момент времени. Именно так в некоторых экспериментах поступают роботы лаборатории робототехники НИЦ "Курчатовский институт". На роботах расположены ИК-маяки, выдающие в частотой 5-10 Гц десятиразрядные коды, определяющие идентификатор робота, его характеристики и код выполняемого в текущий

момент времени действия (состояние). Т.е. робот сообщает явно, что он "спит", "ищет пищу", "убегает" и т.п. Этот подход может иметь и некоторое биологическое основание. Итак, каждое действие имеет свое внешнее проявление.

Вопрос обусловленности подражания несколько сложнее. Будем считать, что элементы системы управления, образующие сеть, снабжены еще одним, помимо возбуждающего или инициирующего, дополнительным подтверждающим входом. Это – вход для сигнала от вершины С.Я. Так, действие не будет активировано, если не будет подтверждающего сигнала от С.Я., интерпретируемого как "принадлежность" этого действия агенту. В определенном смысле это – чувство (ощущение) самости, т.е. отождествление или восприятие объекта, как своего. Без такого ощущения в животном мире происходит рассогласование деятельности. Например, известно сложное психоневрологическое расстройство, называемое синдромом чужой руки. Одним из его клинических симптомов является наличие субъективных ощущений у пациента чужеродности конечности.

Ответ на вопрос "кому подражать" не является очевидным. Речь идет о том, что при наблюдении контрагента происходит не просто определение степени его похожести или близости к агенту. В простейшем случае так работает запаховая метка у муравьев, которые могут определить эту степень близости: от принадлежности контрагента к своему клану до определения его как совершенно чужого. Важнее то, что происходит отождествление наблюдаемого объекта (контрагента) с С.Я. Это означает, что при наблюдении "близкого" контрагента происходит то же подтверждение самости, что и при активизации вершины С.Я. Как происходит это определение степени "похожести" – не существенно. Похожесть чаще всего определяется набором наблюдаемых признаков. Итоговая схема модели подражательного поведения приведена на Рис. 6.

Пунктирная стрелка, ведущая к элементу "Сигнал", означает, что состояние вершины-источника может быть зарегистрировано внешним наблюдателем. Вершина "Наблюдение" отвечает за определение близости наблюдаемого контрагента. На этой схеме вершины s_i отвечают за наблюдаемый набор признаков, приводящих к активизации вершин-детекторов ("Опасность", "Угроза", "Пища") и далее – соответствующих процедур ("Бегство", "Нападение", "Транспортировка").

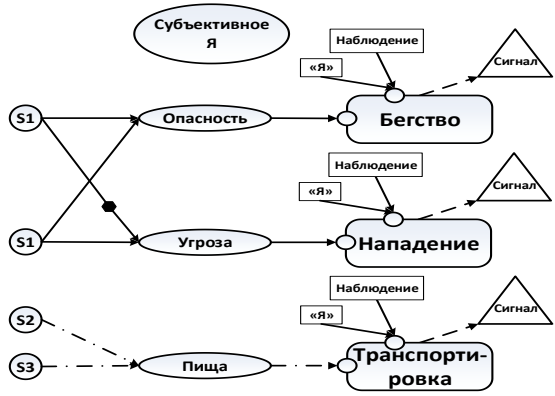


Рис. 6. Пример схемы подражательного поведения. Сплошные стрелки – это сформировавшиеся связи. Штрих-пунктирные стрелки – это формирующиеся в ходе подражания связи

Предположим, что у агента нет связей между наблюдаемыми признаками s_2, s_3 и выполнением действия "Транспортировка". Если агент видит s_2, s_3, а также то, что контрагент совершает действие "Транспортировка" (активен вход "Наблюдение"), то мы получаем систему с активными элементами, между которыми формируется ассоциативная связь. Агент приобретает новый навык.

Подчеркнем еще раз, что основное отличие контагиозного поведения от подражательного заключается именно в приобретении новых навыков, причем это становится возможным в силу того, что в схеме управления определяется вершина С.Я.

2. Социальное обучение

Компоненты С.Я. и "Наблюдение" позволяют описать такой интересный и важный феномен поведения, как социальное обучение (или обучение на чужих примерах). Социальное обучение – это термин, который используется в этологии по отношению к способностям животных приобретать опыт, связанный с взаимодействием с другими особями [Резникова, 2004]. Среди множества видов и форм проявления социального обучения в настоящей работе нас интересует то, которое связано с формированием условных рефлексов. Таких, например, которые наблюдаются в экспериментах с цыплятами: цыплята избегают пищевых единиц характерного вида, если эта пища вызывала реакции отвращения у их сородичей. На самом деле здесь речь идет о т.н. социальной передаче избегания у цыплят. Эффект выглядит так. Новорожденным цыплятам

предъявляют бусину, смоченную жгучим веществом. Поскольку бусина оказывается жгучей на вкус, клюнувшие ее цыплята демонстрируют аверсивную поведенческую реакцию и в дальнейшем, естественно, отказываются клевать такие бусины. Однако замечено, что если другой цыпленок наблюдает процесс обучения, то далее он также начинает избегать клевания таких бусин. У цыпленка-наблюдателя образуется рефлекс. Аналогичные эффекты наблюдаются и у мышей, см., например, [Ивашкина и др., 2019].

Рассмотри схему организации этого эффекта. Фактически, здесь речь идет о некоторой модификации известной схемы формирования условного рефлекса. Когда контрагент клюет бусину, происходит оценка результатов этого действия. Результат действия оценивается и, в зависимости от него, изменяется величина связи между стимулом (бусина s_i) и действием (клевание, a_j). Например, в рамках автоматных моделей М.Цетлина меняется вероятность перехода p_{ij}: при положительной оценке (поощрение) p_{ij} увеличивается, при негативной (наказание) p_{ij} уменьшается [Цетлин, 1969].

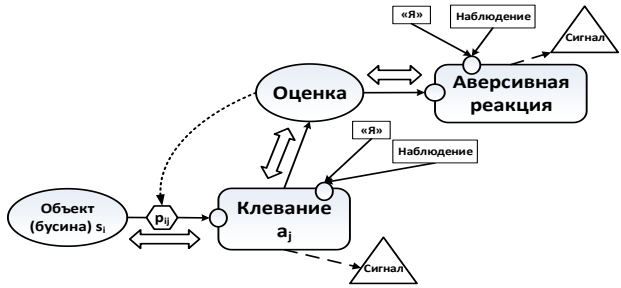


Рис. 7. Формирование реакции избегания. Двойными стрелками показано, что связь между компонентами является фактически двунаправленной

Введение явной компоненты С.Я, а также компоненты "Распознавание" позволяют описать процесс формирования такого же рефлекса и у наблюдателя. Это происходит из-за фактической замены сигнала "Я" наблюдаемым образом. Наблюдение за аверсивной реакцией контрагента в случае его отождествления с С.Я приводит к активизации вершины "Оценка", которая, в свою очередь, изменит вероятность p_{ij} (силу связи) между стимулом и действием. Т.е. образуется аналогичный рефлекс и у наблюдателя. Вторым важным моментом является необходимость признания наличия двунаправленных (взаимных) связей между компонентами. Схема на Рис. 7 – это типичная "функциональная" схема, описывающая направления движение информации, так, как это делается в классической теории управления. Однако переносить такой подход на

процессы, проистекающие в нервной системе, не совсем корректно: связи между нейронами или ансамблями нейронов имеют двунаправленный характер.

Такой подход к формированию социального рефлекса объясняет, в частности, разницу между собственным опытом и тем, который был приобретен в результате обучения (обучение на ошибках других). Собственный опыт является более устойчивым, прочным и продолжительным во времени. Чужой же забывается гораздо быстрее. Это объясняется тем, что активность компонент "Я" и "Наблюдение" разная.

Здесь важно отметить, что речь идет поведении, отличном от подражательного и тем более – контагиозного. Отличие это заключается в том, что здесь фигурирует компонента "Оценка", отсутствующая в схеме подражательного поведения.

3. Вопросы этики поведения

Есть все основания полагать, что наличие С.Я., как явной компоненты, понимание сути процессов отождествления С.Я. с наблюдаемыми контрагентами, а также наличие механизма подражания, – все это открывает путь к пониманию понятия морали поведения агентов с конструктивной точки зрения.

Следуя [Гусейнов, Апресян, 2000], будем считать, что слова "этика", "мораль", "нравственность" будут употребляться как взаимозаменяемые. Разумеется, моральная философия не рассматривает мораль, лишь как механизм групповой и индивидуальной адаптации посредством социальной организации поведения [Апресян, 2017]. Такой "механистический" подход более присущ эволюционным и этологическим теориям, однако, решая сугубо технические задачи, мы вынуждены придерживаться именно "адаптивного" понимания сути морали. При этом мы допускаем применимость моральных оценок к поведению искусственных агентов.

Далее определим три основных вопроса, определяющих содержание морального поведения: (1) зачем такое поведение нужно, (2) какова целевая функция (или основной регулятив), определяющая поведение, (3) каковы механизмы, лежащие в его основе.

1. Необходимость морали. Уже говорилось, что мораль – это механизм адаптации, то, что позволяет функционировать социуму более эффективно. При этом важнейшей особенностью морали является ее гибкость, вариативность. Эта надстройка над базовыми моделями поведения чрезвычайно "легковесна" и может варьироваться в широких пределах на протяжении жизненного цикла индивида.

2. Целевая функция регулирования поведения. Основополагающим регулятивом в межличностных отношений является т.н. "золотое правило" морали. Суть этого правила в определении отношений, основанных на практике взаимности ("ты – мне, я – тебе"). Генезис этого правила описан в работе [Апресян, 2013] и является отдельной темой. Здесь важно лишь отметить, что, во-первых, это правило задает целевую функцию морального поведения. А во-вторых, золотое правило может быть дано в позитивной (поступать по отношению к другим так, как желаете, чтобы поступали по отношению к вам) и негативной (непричинение вреда другим) форме. Для реализации этого основного правила морального поведения требуется определение того, что такое "хорошо" и что такое "плохо" для индивида. Ответ на этот вопрос кроется в понимании сути механизмов, определяющих основу морального поведения.

3. Механизмы морального поведения. Очевидно, что подражательное поведение и социальное обучение можно рассматривать лишь как очень далекий базис для поведения, оцениваемого с точки зрения морали. Однако существует механизм, имеющий более явную связь с этикой поведения и называющийся эмпатией.

Эмпатия. Под этим термином понимается способность к отзывчивости на эмоциональные состояния окружающих индивидов разной степени близости. Разумеется, предрасположенность индивида эмпатии является необходимым, но не достаточным свойством для моральности индивида. Но в этом вопросе нас интересуют два аспекта эмпатии – механизм ее реализации, а также объект эмпатии.

Реализация механизма эмпатии возможна на том же принципе отождествления (или определения степени близости) наблюдаемого агента с С.Я. С объектом эмпатии, как отзывчивости на эмоциональное состояние, дело обстоит несколько сложнее. В определенном смысле эмоции – это, прежде всего, способ интегральной оценки состояния индивида (см. Информационную теорию эмоций П.Симонова, [Симонов, 1982]). Роль эмоций, как фактора, стабилизирующего поведение, контрастирующего сенсорное восприятия и пр. для деятельности не только животных, но и искусственных агентов (роботов), описана, например, в [Karpov, 2014]. Здесь же важно, что эмпатия – это основа для более высокого уровня управления, связанного с целеполаганием и планированием. В терминах моральной философии это означает действие золотого правила: либо реализовывать такой план действий, при котором контрагенту будет хорошо (увеличение уровня эмоционального состояния, позитивная формулировка правила), либо сформировать план, не ведущий к появлению отрицательных эмоций контрагента (негативная формулировка). В любом случае эмоциональное состояние контрагента оказывает влияние на формирование мотива поведения, его цели.

Таким образом, мы можем постулировать, что: (1) в перечень механизмов, определяющих нравственное поведение, наряду с прочими, входят социальное обучение и эмпатия и (2) основной мотивацией морального поведения (то, на что направлено золотое правило морали) является максимизация эмоционального уровня контрагента, на которого направлено воздействие индивида. Поскольку знак и величина эмоции непосредственно определяется существующими потребностями агента, то можно сделать весьма "механистический" вывод: действие основного морального регулятива направлено на удовлетворения потребностей индивида.

Такая неизбежная вульгаризация – это результат попытки переноса сугубо технических механизмов (вплоть до адаптационных моделей) на область, принципиально плохо формализуемую – моральную философию.

Заключение

Итак, было показано, что введение понятия С. Я. как конструктивного элемента системы управления позволяет предложить ряд моделей, реализующих такие важнейшие компоненты социального взаимодействия, как подражательное поведение и социальное обучение. Кроме того, эта компонента является конструктивной и в вопросе моделирования такой способности социального индивида, как эмпатия. Однако попытки механического переноса этих механизмов для объяснения или обоснования моральных принципов организации взаимодействия, даже как адаптационного механизма, логически приводят к весьма примитивным и банальным выводам. Отчасти причина тому – отсутствие общего понятийного аппарата.

В любом случае, если продолжать исследование аспектов моральности поведения аниматов "снизу", с "технической" стороны, то остается открытым целый ряд вопросов, таких, например, как:

1. Насколько нравственные правила определяются подражательным поведением, социальным обучением и способностью к эмпатии?
2. Насколько целесообразно интерпретировать формы поведения анимата именно с точки зрения основного нравственного правила?
3. Что подлежит регуляции в "моральном поведении"? Только ли характеристика близости "свой-чужой"?
4. Какова значимость фактора необходимости совместной деятельности для решения сложных задач в зависимости от свойств среды обитания и индивидуальных потребностей?

Эти вопросы требуют своего разрешения, причем вне зависимости от положений моральной философии, а оставаясь в рамках парадигмы моделей социального поведения роботов.

Список литературы

[Karpov, 2014] Karpov V. Robot's temperament // Biol. Inspired Cogn. Archit. 2014. V. 7. С. 76–86.

[Апресян, 2013] Апресян Р.Г. Генезис золотого правила // Вопросы философии. 2013. № 10. С. 39–49.

[Апресян, 2017] Апресян Р.Г. Этика: учебник. М.: КНОРУС, 2017. 356 с.

[Гусейнов, Апресян, 2000] Гусейнов А.А., Апресян Р.Г. Этика. М.: Гардарики, 2000. 472 с.

[**Ениколопов, Кузнецова, Чудова, 2014]** Ениколопов С.Н., Кузнецова Ю.М., Чудова Н.В. Агрессия в обыденной жизни. М.: Полит. энциклопедия, 2014. 496 с.

[Знаков, 2005] Знаков В.В. Психология понимания: Проблемы и перспективы. М.: «Институт психологии РАН», 2005. 448 с.

[Зорина, Полетаева, Резникова, 2002] Зорина З.А., Полетаева И.И., Резникова Ж.И. Основы этологии и генетики поведения. Учебник. 2-е изд. М.: Изд-во МГУ: «Высшая школа», 2002. 383 с.

[Ивашкина и др., 2019] Ивашкина О.И. и др. Социальная передача страха у мышей: влияние прошлого индивидуального обучения в задаче условно-рефлекторного замирания // Всероссийская с международным участием Конференция: XLIV Итоговая научная сессия «Системная организация физиологических функций». М., 2019.

[Карпов, Карпова, Кулинич, 2019] Карпов В.Э., Карпова И.П., Кулинич А.А. Социальные сообщества роботов. М.: УРСС, 2019. 352 с.

[Карпова, Карпов, 2018] Карпова И.П., Карпов В.Э. Агрессия в мире аниматов, или О некоторых механизмах управления агрессивным поведением в групповой робототехнике // Управление большими системами. 2018. Т. 76. С. 173–218.

[Кулинич, 2018] Кулинич А.А. Модель командного поведения агентов в качественной семиотической среде. Часть 2. Модели и алгоритмы формирования и функционирования команд агентов // Искусственный интеллект и принятие решений. 2018. № 1. С. 29–40.

[Лоренц, 1994] Лоренц К. Агрессия (так называемое «зло»). М.: Республика, 1994. 272 с.

[Лурия, 2007] Лурия А.Р. Лекции по общей психологии. Питер, 2007. 320 с.

[Осипов и др., 2018] Осипов Г.С. и др. Знаковая картина мира субъекта поведения. М.: Физматлит, 2018. 264 с.

[Резникова, 2004] Резникова Ж.И. Сравнительный анализ различных форм социального поведения у животных // Журнал общей биологии. 2004. Т. 65. № 2. С. 135–151.

[Симонов, 1982] Симонов П.В. Потребностно-информационная теория эмоций // Вопросы психологии. 1982. Т. 6. С. 44–56.

[Тинберген, 1993] Тинберген Н. Социальное поведение животных. М.: Мир, 1993. 81 с.

[Цетлин, 1969] Цетлин М.Л. Исследования по теории автоматов и моделированию биологических систем. М.: Наука, 1969. 316 с.

УДК 34.025

БЕСПИЛОТНЫЕ ТРАНСПОРТНЫЕ СРЕДСТВА КАК НОВЫЙ ПРЕДМЕТ РЕГУЛИРОВАНИЯ В СОВРЕМЕННЫХ ГОСУДАРСТВАХ, МЕЖДУНАРОДНЫХ ОРГАНИЗАЦИЯХ И МЕЖДУНАРОДНЫХ ИНТЕГРАЦИОННЫХ ОБЪЕДИНЕНИЯХ

А.Ж. Степанян (*armen@stepanyan.com*)
Московский государственный университет
имени О. Е. Кутафина, Москва

Аннотация. Для понимания эффективности и системы регулирования беспилотных транспортных средств необходимо определить объект регулирования. В статье выделяются аспекты правоотношений, возникающих при использовании БТС? Которые стоит учесть законодателю, а также анализируется текущее правовое регулирование этой сферы на международном, интеграционном и национальном уровнях.[1]

Ключевые слова: беспилотные транспортные средства, БТС, правовое регулирование.

Введение

По данным газеты The Economist, 90% дорожно-транспортных происшествий происходят из-за ошибок человека . Достаточно логично предположить, что аварийность можно свести к нулю, если заставить человека не ошибаться либо убрать человека с места водителя. Первое сделать невозможно, причиной этому является сущность человека: errare humanum est. Но можно ли убрать человека как водителя-пилота ?

Представляется, что технически этот вопрос будет решен на обычном для обывателя уровне через 1-2 года за рубежом и 3-5 лет в России, а разница обусловлена сложным климатом, недисциплинированностью дорожных служб и невысокой законопослушностью водителей-людей. Более сложной задачей технически являются отсутствие пилота в транспортных средствах в воздухе: самолетах, вертолетах. Создается также целый класс машин, заведомо не могущих иметь пилота в силу своего

[1] Работа выполнена при финансовой поддержке РФФИ (проект № 18-29-16172).

размера и конструкции. Повышенный интерес в таких областях как военная и правоохранительная деятельность, промышленность с тяжелыми и опасными для человека природными условиями работы представляют транспортные средства, в которых также отсутствует человек.

Вопросы регулирования использования БТС

Сущностно транспортное средство без водителя как относительно новое явление, объект ставит перед законодателем множество вопросов. Во-первых, это роль искусственного интеллекта (далее – ИИ) в управлении. Это проявляется даже терминологически. По-английски синонимичны понятия driverless car (машина без водителя), autonomous car (автономная машина), robotic car (роботизированная машина) и self-driving car (самодвижущаяся машина или самоуправляемая машина). Но они существенно отличаются по смыслу, по действиям, по возможностям ИИ и человека. Различие в стилистике словосочетания подчеркивает эти отличия. Само по себе слово «автономия» уже говорит о степени свободы «воли машины» (что само по себе неоднократно поднималось и будет подниматься в литературе).

Во-вторых, это степень ИИ в управлении транспортом: фактически сегодня самолеты и автомобили являются полуавтономными средствами. Круиз-контроль, автопилот, слежение за полосой. По сути, полуавтономные средства оперируют самостоятельно частью функций: разгон, торможение, - но требуют надзора и контроля со стороны водителя, имеющего право на управление. Стоит не забывать, что водительские права в Российской Федерации являются документом о специальном праве, наделяемом государством только после сдачи экзамена. По своей сути они являются лицензией в смысле административного права, что прямо закреплено в законодательстве некоторых стран, например, США.

По уровням автоматизации в настоящее время в США и в других странах (Китай, Нидерланды и другие) используется классификация SAE :

Уровень 0 – нет автоматизации. Постоянное выполнение человеком-водителем всех аспектов динамического вождения, даже если оно улучшено системами предупреждения или вмешательства.

Уровень 1 – помощь водителю: осуществление системой в зависимости от режима вождения помощи водителю либо рулевого управления, либо ускорения (замедления) с использованием информации об окружающей среде вождения и с ожиданием того, что водитель-человек выполнит все остальные аспекты задачи динамического вождения.

Уровень 2 – частичная автоматизация: осуществление в зависимости от режима вождения одной или несколькими системами помощи водителю как в рулевом управлении, так и в ускорении (замедлении) с использованием

информации об окружающей среде вождения и с учетом того, что водитель-человек выполняет все остальные аспекты задачи динамического вождения.

Уровень 3 – условная автоматизация: с помощью автоматизированной системы вождения производительность, характерная для соответствующих режимов вождения, по всем аспектам задачи динамического вождения с расчетом на то, что водитель-человек соответствующим образом ответит на запрос о вмешательстве.

Уровень 4 – высокая автоматизация: эффективность режима вождения с помощью автоматизированной системы вождения во всех аспектах задачи динамического вождения, даже если водитель-водитель не отвечает должным образом на запрос о вмешательстве.

Уровень 5 – полная автоматизация: постоянная работа автоматизированной системы вождения по всем аспектам задачи динамического вождения при любых дорожных условиях и условиях окружающей среды, которой может управлять человек-водитель.

Представляется важным отметить то, что даже в США после представления своей попытки классификации была выбрана классификация отраслевой организации [Glancy, 2017].

Какие вопросы встают еще перед создателями законодательства о беспилотных транспортных средствах?

Одним из самых популярных вопросов является «Кто несет гражданскую ответственность в случае ДТП с участием беспилотного транспортного средства?» Это также один из важных вопросов с точки зрения этики, ибо он затрагивает категорию справедливости, а также вопрос равенства машин и людей [Rapaczynski, 2016] [Kamala, 2016]. С точки зрения конституционной экономики развитый рынок страховых услуг должен помочь решить этот вопрос, сделав нетрудным фактический ответ на него. В этом также должно помочь развитое законодательство. Одним из возможных решений видится введение режима, подобному режиму «источника повышенной опасности»: тогда в ДТП ответственность будут разделять все участвующие в нем стороны. Но такое решение представляется далеко не лучшим с точки зрения морали и этики и, скорее, является показательно «капиталистическим».

Вопросом в том числе этики является вопросы выдачи разрешений на использование таких средств, ведь право на вождение является специальным правом, по сути своей прерогативой лишь определенных лиц. Разрешенное использование полной автоматизации можно трактовать и как дозволение определенных механизмов, но и как дарование равного с определенными квалифицированными людьми права машине. И если для людей выработаны достаточные критерии такой квалификации, то пока еще таких критериев для машины нет [Gurney, 2016].

В процессе перемещения транспортные средства будут собирать большое количество данных как об объектах снаружи, так и внутри салона. Рискну предположить, что как минимум первое время производители будут фактически нарушать право на частную жизнь, собирая данные, не относящиеся напрямую к улучшению статистики вождения, хотя и могущие затронуть ее. Достаточно многие автолюбители произносят целые монологи, передвигаясь по дороге, в том числе относящиеся к удобству вождения именно в этом автомобиле.

Представляется также скорым изменение законодательства о градоустройстве и строительстве с целью развития стандартов, адаптированных в том числе для беспилотных средств. Необходимым будет также либерализация законодательства в области частот, столь необходимых для взаимодействия транспортных средств друг с другом, а также с инфраструктурой пользователей и путей сообщения .

Уровень регулирования использования БТС

Важнейшим вопросом является уровень регулирования использования автономных, полуавтономных транспортных средств. национальный уровень, интеграционный или международный.

Венская конвенция по дорожному движению 1968 года , сторонами которой являются 78 государств, в пункте 1 статьи 8 указывается, что каждое транспортное средство или состав транспортных средств, которые находятся в движении, должны иметь водителя. При этом пункт 5 требует, чтобы водитель был всегда в состоянии управлять своим транспортным средством. Данный пункт фактически сводит на нет преимущества автономного самоуправления для водителя. Что примечательно, Норвегия, ратифицировавшая эту конвенцию лишь в 1985 году, сделала оговорку именно в отношении пункта 5 статьи 8 . В 2014 году в статью 8 были внесены изменения, позволяющие автомобилю ехать самому, если система «может быть переопределена или выключена водителем», то есть, по сути, хотя и открывающая возможность автономной езды, но под присмотром человека, то есть, никак не без водителя.

На национальном уровне представлено широкое разнообразие форм регулирования. Возглавляет это разнообразие США с отсутствием компетенции федерации в области роботизированного движения и законами, постановлениями и даже просто анонсами инициатив по разрешению использования беспилотных транспортных средств на дорогах общего пользования. Самым продвинутым штатом является Калифорния с его правилами , разрешающими полное автономное вождение на дорогах общего пользования. Первым штатом, принявшим законодательство о беспилотных транспортных средствах, была Невада, за ним Аризона.

Китай представляет действующие на уровне самых развитых городов (Пекин, Шанхай, Шэньджэнь и другие) правила испытаний для автономных автомобилей по дорогам общего пользования. Германия – на уровне земель. Нидерланды также приняли закон о тестовой эксплуатации автономных транспортных средств в 2017 году . Южная Австралия, Сингапур, Япония и множество других регионов и стран идут тем же путем. Новая Зеландия утвердила тесты летающего такси .

Несмотря на то, что в передовых областях Европейский Союз старается принять законодательство или хотя бы осветить свое видение политик регулирования раньше, чем начинается повсеместное использование технологий обывателями, в настоящее время не принято ни одного акта, ни «книги», лишь в области управления транспортом и другими. Представляется необходимым, чтобы ЕС сделал публичной свою позицию относительно возможности использования беспилотных средств с элементами искусственного интеллекта.

Заключение

Как видно из обзора действующего регулирования, национальное регулирование достаточно фрагментарно, международное как таковое отсутствует и отстает от национального. Именно поэтому интеграционные объединения – Европейский Союз и Евразийский Экономический Союз могут, проявив законодательную инициативу даже в формах мягкого права (soft law), обеспечить рост экономики и развитие транспортного и логистического сектора путем принятия актов, направленных на всестороннюю поддержку использования беспилотных транспортных средств на своей территории..

Список литературы

[Glancy, 2017] Glancy, Dorothy, What Will the Law Do About Autonomous Vehicles? 23/05/2013. // http://dx.doi.org/10.2139/ssrn.2293051, Research, AAJ, Driven to Safety: Robot Cars and the Future of Liability 07.02.2017. // http://dx.doi.org/10.2139/ssrn.2913028, (дата обращения 02.03.2019)

[Gurney, 2016] Gurney, Jeffrey, Crashing into the Unknown: An Examination of Crash-Optimization Algorithms Through the Two Lanes of Ethics and Law (March 8, 2016). 79 Albany Law Review 183 (2016). // https://ssrn.com/abstract=2622125, (дата обращения 02.03.2019)

[Kamala, 2016] Kamala Kelkar, How Will Driverless Cars Make Life-or-Death Decisions? PBS 28/05/2016, // http://www.pbs.org/newshour/rundown/how-will-driverless-cars-make-life-or-death-decisions/, (дата обращения 02.03.2019)

[Rapaczynski, 2016] Rapaczynski, Andrzej, Driverless Cars and the Much Delayed Tort Law Revolution 14.04.2016. Columbia Law and Economics Working Paper No. 540. // http://dx.doi.org/10.2139/ssrn.2764686 (дата обращения 02.03.2019)

УДК 34.025

АНАЛИЗ ТЕРМИНОЛОГИЧЕСКИХ И СОДЕРЖАТЕЛЬНЫХ АСПЕКТОВ ПОНЯТИЙ «ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ» И «РОБОТОТЕХНИКА» В СВЕТЕ НЕОБХОДИМОСТИ ИХ ПРАВОВОГО РЕГУЛИРОВАНИЯ

К.С. Яковлев (*yakovlev@isa.ru*)
Федеральный исследовательский центр «Информатика и управление» Российской академии наук, Москва

А.В. Боковой (*bokovoy@isa.ru*)
Федеральный исследовательский центр «Информатика и управление» Российской академии наук, Москва

С.Ю. Кашкин (*info@eulaw.edu.ru*)
Московский государственный университет имени О. Е. Кутафина, Москва

Аннотация. На текущий момент, в России отсутствует фундаментальное правовое исследование на тему регулирования статуса роботов и систем искусственного интеллекта в системе правового регулирования современных государств, международных организаций и международных интеграционных объединений. Целью работы является анализ терминологических и содержательных аспектов вышеописанных понятий в стандартах и правовых документах стран и международных объединений с развитой системой регулирования в областях робототехники и искусственного интеллекта.[1]

Ключевые слова: Искусственный интеллект, робототехника, правовое регулирование, стандарты, зарубежный опыт, Российская Федерация.

Введение

В настоящее время наблюдается активное развитие робототехнических систем, в том числе способных к функционированию в автономном режиме,

[1] Работа выполнена при финансовой поддержке РФФИ (проект № 18-29-16150).

т.е. без участия человека, при определенных обстоятельствах. Если раньше подобные системы применялись в основном на производстве, в военной сфере, космосе, то сейчас их можно встретить и в повседневной жизни. Отметим в этой связи, например, автомобиль «Tesla S» [Delphi, 2014] с системой автономного управления (автопилотом), складских коллаборативных роботов, работающих вместе с человеком от компании Amazon Robotics [Porter, 2018], робота-хирурга «Da Vinci» [Maeso et al., 2010] и др. При этом, как в России, так и во всем мире не существует законов, которые в полной мере покрывали бы случаи использования робототехники и искусственного интеллекта в реальных условиях, что оставляет вопросы о причастности кого-либо и тяжести вины при нарушении законодательства подобными устройствами и интеллектуальными системами открытыми. В связи с этим, России необходима правовая база, которая позволит эффективно осуществлять регулирование в областях робототехники и искусственного интеллекта. Для успешного создания такой правовой базы, в качестве одного из этапов должен быть проведен терминологический и содержательный анализ таких понятий как «робототехника» и «искусственный интеллект» с учетом правового опыта, накопленного современными государствами, международными организациями и международными интеграционными объединениями.

1. Термин «робототехника»

Одним из основных источников, претендующих на объективное и точное раскрытие термина «робототехника» и связанных с ним понятий, являются международные и национальные стандарты. Стоит отметить стандарт ISO 8373:2012 [ISO, 2018], в котором ISO четко разграничиваются такие понятия «манипулятор» – механизм, состоящий из нескольких звеньев и позволяющий осуществлять перемещение предметов, и «робот» – приводной механизм, обладающий некоторой степенью автономности. Под автономностью в стандарте обозначена способность некоторого устройства выполнять целевые задачи, основываясь на текущем состоянии устройства и полученных данных без вмешательства человека. Заметим при этом, что понятие «обладающий» не уточняется, что может привести к коллизиям. Рассмотрим, например, современный мультироторный беспилотный летательный аппарат (коптер, дрон) от компании DJI [Schroder et al., 2015]. Этот (и подобные ему) летающие роботы с одной стороны подразумевают управление человеком-оператором, т.е. являются манипуляторами с точки зрения стандарта, с другой – имеют возможность осуществлять автономные полет и облет препятствий по предварительно составленному маршруту в виде последовательности GPS-координат. Очевидно, что в момент такого

неконтролируемого человеком полета аппарат можно отнести к «роботам» по стандарту.

Формально термин «робот» в стандарте ISO означает приводной механизм, программируемой по нескольким осям, имеющий некоторую степень автономности, движущийся внутри рабочей среды и выполняющий задачи по предназначению. При этом, в документе разделяются промышленные и обслуживающие роботы в зависимости от их предназначения. Также, в стандарте приводится классификация роботов по типу механических конструкций: колесный, шагающий, двуногий, гусеничный или рельсовый, антропоморфный, мобильный. При этом пропущена четкая классификация воздушных, наводных и подводных роботов, которые могут выполнять схожие с наземными роботами задачи. На основе термина «робот» дано определение термина «робототехника» как науки и практики разработки, производства и применения роботов. В целом, несмотря на то, что указанный стандарт был принят относительно недавно, на текущий момент его с содержательной точки зрения можно считать устаревшим. Очевидно это понимание есть и у самой организации ISO, которая ведёт в настоящий момент разработку нового стандарта, который заменит ISO 8373:2012, а именно - ISO/CD 8373 Robotics – Vocabulary.

В стандарте IEEE 1872-2015 Ontologies for Robotics and Automation [IEEE, 2015] от авторитетной в области робототехники организации IEEE робот определяется как агентное устройство в широком смысле, предназначенное для функционирования в реальном мире для выполнения одной или более задач (англ. «An agentive device in a broad sense, purposed to act in the physical world in order to accomplish one or more tasks»). При этом для роботов выделятся следующие *роли*:

- automated robot – роль, когда робот действует по принципу автомата, т.е. по заданной программе и не адаптируется к окружающей среде

- fully autonomous robot – роль, когда робот выполняет поставленную задачу без какого-либо управления со стороны человека, адаптируясь при этом к изменениям в окружающей среде;

- semi-autonomous robot – роль, когда робот и человек совместно планируют выполнение поставленной задачи и осуществляют исполнение плана, варьируя степень участия человека;

- teleoperated robot – роль, когда робот контролируется оператором, который может задавать как непосредственно управляющие воздействия на исполнительные механизмы робота, так и промежуточные цели (incremental goals), которые робот может достигать самостоятельно;

- remote-controlled robot – роль, когда робот полностью контролируется оператором, находящимся вне робота.

На наш взгляд, идея ролей весьма разумна, т.к. такой подход допускает, что один и тот же робот может как управляться человеком, так и

осуществлять самостоятельные действия (т.е. быть автономным), в зависимости от возникающих задач. Однако при этом, сами определения ролей достаточно размытые и четкой границы между ними провести нельзя. Например, достаточно проблематично точно определить различие между teleoperated и remote-controlled. Роль же semi-autonomous описана настолько широко, что почти любого робота можно атрибутировать ею.

На национальном уровне, в России, принят стандарт ГОСТ Р ИСО 8373-2014 «Роботы и робототехнические устройства. Термины и определения» действующий с 01.01.2016 г и переизданный в январе 2019 г. Этот стандарт идентичен международному стандарту ISO 8373:2012, который, как было сказано выше, можно считать устаревшим.

2. Термин «искусственный интеллект»

В отличие от рассмотренной выше области робототехники, в искусственном интеллекте не так развита система стандартов. Очевидно, это связано с тем, что искусственный интеллект, зародившись как научное направление, долгое время был предметом интереса лишь академического сообщества. В этой связи представляется разумным проанализировать не только имеющиеся попытки терминологически описать ИИ на уровне стандартов, но и уделить внимание академическим определениям.

Сам термин «искусственный интеллект» был введён в научный обиход Джоном Маккарти в 1956 году. В этом году был организован первый семинар по искусственному интеллекту. В заявке на проведение этого семинара говорилось о том, что основной целью является поиск способов «заставить» (англ. make) машины использовать язык (имеется в виду – естественный), формировать абстракции, решать задачи свойственные человеку, улучшать себя. Как мы видим, это достаточно смелая цель для 1956 года, которая не решена до сих пор, но она может являться некоторым контуром, очерчивающим концептуальное содержание искусственного интеллекта, как научного направления. Через некоторое время после семинара на регулярной основе стала проводиться международная конференция по искусственному интеллекту – IJCAI (International Joint Conference on Artificial Intelligence). Она проводится до сих пор. Индуктивно, можно считать, что те вопросы, которые на ней рассматриваются и составляют предмет искусственного интеллекта.

Активными игроками в академической среде являются национальные ассоциации искусственного интеллекта. Наиболее крупной является AAAI – Association for Advancement of Artificial Intelligence (бывшая американская ассоциация ИИ). В России функционирует РААИ – Российская ассоциация искусственного интеллекта. Являясь членом РАИИ, один из авторов не понаслышке знаком с теми трудностями, которые испытывают члены

ассоциации при попытке дать непротиворечивое определение термину «искусственный интеллект». Какое-то время назад ассоциацией была инициирована попытка зафиксировать содержание этого термина, однако она пока не реализована. Частные же определения имеются в достаточном количестве. Наиболее подходящее, на наш взгляд, принадлежит президенту РАИИ Г.С. Осипову. «Научная дисциплина «искусственный интеллект» входит в область компьютерных наук. Основной целью исследований в искусственном интеллекте является получение методов, моделей и программных средств, позволяющих искусственным устройствам реализовывать целенаправленное поведение и разумные рассуждения» [Осипов, 2011]. Зафиксировав это определение, как наиболее удачное, по нашему мнению, перейдем к анализу имеющихся инициатив по стандартизации в области ИИ.

Одной из наиболее масштабных инициатив является работа технического комитета ISO/IEC JTC 1/SC 42. Эта инициатива объединяет 22 страны в статусе «участник» (в том числе Россию) и 14 стран в статусе «наблюдатель». Ведётся работа над стандартом ISO/IEC WD 22989 «Artificial intelligence – Concepts and terminology», который, очевидно, будет включать в себя определение(я) искусственного интеллекта. На текущий момент (первый квартал 2019) документ находится в стадии working draft (черновик), т.е. открытого доступа к имеющимся определениям нет.

Еще одной глобальной инициативой для изучения этических аспектов автономных и интеллектуальных систем является «The IEEE Global Initiative on Ethics of Autonomous and Intelligent Systems», в которую входят 14 рабочих групп из разных стран мира. На текущий момент, основной работы инициативной группы является сборник основных проблем и практических рекомендаций для правительства, научных организаций и бизнеса, использующих автономные и интеллектуальные системы, «Ethically Aligned Design, First Edition». В работе, помимо прочего, выделены проблемы правового регулирования автономных и интеллектуальных систем и использования их в юридических системах. Среди основных проблем, относящихся к этому разделу, можно выделить следующие:

- улучшения, которые могут внести интеллектуальные системы в быт человека;
- препятствия к информационному доверию;
- могут ли улики и доказательства, полученные интеллектуальной системой, быть использованы в правовой системе;
- как уточнение знаний и навыков оператором влияет на доверие и пригодность интеллектуальных систем в правовых системах;
- прозрачность рассуждений интеллектуальных систем в процессе принятия решений.

Большинство правовых актов, применяемых сегодня для регулирования ИИ и робототехники по своей юридической силе не являются обязательными и представляют собой рекомендации, декларации, хартии и т.д. Рассмотрим некоторые из них.

Например, Японским обществом по искусственному интеллекту [JSAI, 2017] были разработаны этические стандарты для создателей юнитов ИИ. Эти этические рекомендации должны стать морально-этической базой деятельности членов общества в интересах человека. Они включают в себя гуманизм, соблюдение прав человека, законность, честность, безопасность, ответственность и т.д.

Корейская хартия этических норм для роботов состоит из семи статей [Hilgendorf et Kim, 2012]. Хартия предлагает аналогичные этические стандарты для роботов и людей, целью которых являются совместные блага для человека и машины, взаимное уважение достоинства друг друга, не нанесение ущерба человеку и забота о роботе. При этом правительству подлежит обеспечивать соблюдение этих правил.

Азиломарские принципы искусственного интеллекта явились результатом конференции по ИИ 2017 г. в Азиломаре (Калифорния, США). 23 принципа определяют цели, ценности и преспективы создания и использования ИИ, их гуманное взаимодействие с наукой, культурой и производством во благо человечеству.

Анализ новейших правовых инициатив и решений показывает, что для того, чтобы оградить человека от негативного влияния искусственного интеллекта в сфере морали и нравственности необходим правовой контроль и ответственность за создателями, производителями, владельцами, пользователями, арендаторами юнитов искусственного интеллекта, наносящих ущерб людям.

Добро и возможное зло, исходящие от юнитов искусственного интеллекта должны быть под четким и строгим правовым контролем человека. Их действия необходимо совместить с ответственностью, подотчетностью, а в соответствующих ситуациях – немедленной прекращаемостью.

Поэтому человечеству следует адекватно представлять возможные пути, темпы и последствия этого движения и уметь вовремя остановиться. Отсюда возникает и парадоксальная мысль о необходимости и возможности правового регулирования человеческого инстинкта самосохранения. Надо подумать о коллективном самосохранении человеческого общества в возможном столкновении с искусственным интеллектом и о недопустимости неконтролируемого появления и саморазвития этой способности у интеллектуального робота!

В работе представлены рекомендации по каждому вопросу, которые можно свести к необходимости правового контроля автономных и

интеллектуальных систем правительством, вплоть до контроля обучающих выборок для самообучающихся систем и создание единого стандарта интерпретации рассуждений интеллектуальных систем при принятии решений. В работе отмечается, что решение проблем правового регулирования автономных систем возможно только при тесном сотрудничестве правительства с научными и коммерческими организациями, имеющими большой опыт в разработке, внедрении и эксплуатации таких систем.

3. Необходимость законодательного регулирования

Сегодня, в условиях шестого научно-технологического уклада, под воздействием искусственного интеллекта и робототехники происходит кардинальная смена всего устоявшегося миропорядка. Меняется экономика, политика, право, идеология, общественные отношения, ценности и даже сама человеческая личность. Поэтому право должно стать тем механизмом, который призван отрегулировать эти сложные отношения. Точность и адекватность юридической терминологии имеет в этом инновационном вопросе особое значение.

Изучение правового регулирования искусственного интеллекта и робототехники прошло три этапа. Оно берет свое начало, как уже отмечалось, в США в 50-е годы XX в. В 70-е годы оно приобрело более конкретные формы и первоначально было сконцентрировано в многочисленных университетских научно-образовательных программах и крупных юридических фирмах.

Следующий, второй этап продолжился в 90-е годы, когда интерес к ИИ и робототехнике проявили и другие, прежде всего наиболее высокоразвитые государства. Появились первые правовые акты, регулирующие эту сферу жизни.

С 2015 года начался современный, третий этап, когда вопросы правового регулирования искусственного интеллекта и робототехники стали общепризнанной практической проблемой в большинстве стран мира, а также и в интеграционных объединениях, выйдя на наднациональный и глобальный уровень.

В США, в отличие от ЕС, правовое регулирование ИИ и робототехники, не имеет четкой национальной стратегии и системного характера. В соответствии с унаследованной от Великобритании традицией, американцы весьма эффективно применяют инструменты судебной власти и прецедентного права [Calo, 2016], а также успешно концентрируют усилия на некоторых конкретных направлениях и программах.

Интересно, что правовое регулирование в сфере ИИ и робототехники весьма эффективно используется в восточных странах с иероглифической

письменностью, для которой характерно более «клиповое», «картиночное» мышление (в отличие от европейских языков) и традиции послушания и уважения власти.

Например, в Китае (привыкшем к строгому социалистическому правопорядку) правовое регулирование ИИ и робототехники, более чем в других государствах, продвинулось на современном этапе от необязательных деклараций к вполне четко соблюдаемым, в том числе плановым и финансовым правовым актам. Так 13-й Пятилетний план развития КНР на 2016–2020 гг. предусматривает большой скачок в сфере искусственного интеллекта [КНР, 2016]. На этот же период практически одновременно был принят план развития робототехнической промышленности [КНР2020, 2016]. Планом развития в Китае технологий искусственного интеллекта нового поколения, опубликованным в 2017 г. «поднебесная» официально продемонстрировала стремление стать мировым лидером в этой области [КНР, 2017]. Эту же цель в отношении робототехники к 2030 г. преследует и государственная программа «Сделано в Китае – 2025».

В нашей стране эти вопросы находятся еще в самой начальной стадии изучения и регулирования [Морхат, 2018], а потому имеют особое значение и требуют обстоятельного, системного и комплексного изучения.

Чрезвычайно активное участие в развитии правового регулирования ИИ и робототехники на современном этапе уже на наднациональном уровне принимает Европейский Союз. Так в 2015 г. в число 10 принятых Европейским советом приоритетных направлений деятельности комиссии ЕС были включены искусственный интеллект и робототехника.

25 государств–членов ЕС подписали 10 апреля 2018 г. Декларацию о сотрудничестве в области искусственного интеллекта в соответствии с принципами, целями и ценностями ЕС, а также для защиты прав граждан [ЕС, 2018].

Принятый 1 января 2019 года Проект в области искусственного интеллекта AI4EU явился еще одним серьезным практическим шагом к формированию общеевропейской базы правового регулирования ИИ [ai4eu-project, 2019]. В нем зафиксированы правовые и инвестиционные достижения ЕС в этой области.

Анализ и выводы

Состояние правового регулирования робототехники и искусственного интеллекта на текущий момент в мире находится в зачаточном состоянии. Робототехника и искусственный интеллект стали одной из самых известных научно-технологических тенденций нашего века и одной из реальных инновационных проблем дня сегодняшнего. Уже сегодня разрабатываются,

функционируют и активно применяются юниты искусственного интеллекта. Одной из основных проблем является отсутствие четкой терминологии в упомянутых областях. Для робототехники уже разработан стандарт, частично решающий эту проблему, тогда как для искусственного интеллекта, в том числе как предмета академического интереса, единого стандарта нет, и в настоящее время работа над таким документом только начинается.

Список литературы

[Осипов, 2011] Осипов Г.С. Методы искусственного интеллекта. // М.: ФИЗМАТЛИТ, 2011. – 296 с.

[ai4eu-project,2019]https://ec.europa.eu/digital-single-market/en/news/artificial-intelligence-ai4eu-project-launches-1-january-2019

[Delphi, 2014] Delphi'2014 «Владелец Tesla проехал все США абсолютно бесплатно» - https://www.delfi.lv/avto/na-kolesah/vladelec-tesla-proehal-vse-ssha-EU absolyutno-besplatno.d?id=44084741 (дата обращения – 01.03.2019)

[EC, 2018] EU Member States sign up to cooperate on Artificial Intelligence, European Commission // https://ec.europa.eu/digital-single-market/en/news/eu-member-states-sign-cooperate-artificial-intelligence (дата обращения 01.03.2019).

[IEEE, 2015] IEEE «Standard Ontologies for Robotics and Automation» // IEEE Std 1872-2015 , vol., no., pp.1-60, 10 April 2015

[ISO, 2018] ISO'2018 Robots and Robotic Devices – Vocabulary - https://www.iso.org/obp/ui/#iso:std:iso:8373:ed-2:v1:en (дата обращения – 01.03.2019)

[Maeso et al., 2010] Maeso, S., Reza, M., Mayol, J. A., Blasco, J. A., Guerra, M., Andradas, E., & Plana, M. N Efficacy of the Da Vinci surgical system in abdominal surgery compared with that of laparoscopy: a systematic review and meta-analysis. // Annals of Surgery – 2010 – p. 254-262.

[Porter, 2018] Porter B. «VP of Robotics at Amazon, on Warehouse Automation, Machine Learning, and His First Robot» - https://spectrum.ieee.org/automaton/robotics/industrial-robots/interview-brad-porter-vp-of-robotics-at-amazon (дата обращения 01.03.2019)

[Schroder et al., 2015] Schroder, A., Renker, M., Aulenbacher, U., Murk, A., Boniger, U., Oechslin, R., & Wellig, P. Numerical and experimental radar cross section analysis of the quadrocopter DJI Phantom 2 // 2015 IEEE Radar Conference. – IEEE, 2015. – C. 463-468.

[JSAI, 2017] The Japanese Society for Artificial Intelligence Ethical Guidelines http://ai-elsi.org/wp-content/uploads/2017/05/JSAI-Ethical-Guidelines-1.pdf (дата обращения 01.03.2019)

[Hilgendorf et Kim, 2012] Примерный текст описывается по Legal Regulation of Autonomous Systems in South Korea on the Example of Robot Legislation // Prof. Dr. Dr. Eric Hilgendorf Minkyu Kim. // https://www.jura.uni-wuerzburg.de/fileadmin/_migrated/content_uploads/Legal_Regulation_of_Autonomous_Systems_in_South_Korea_on_the_Example_of_Robot_Legislation_-_Hilgendorf_Kim_05.pdf (дата обращения: 01.03.2019)

[КНР, 2016]The 13th Five-Year Plan For Economic and Social Development of the People's Republic of China (2016–2020), 2016, National Development and Reform Commission (NDRC) PRC // http://en.ndrc.gov.cn/newsrelease/201612/P020161207645765233498.pdf (дата обращения: 01.03.2019)

[КНР2020, 2016] План развития робототехнической промышленности (2016-2020) (2016), текст на китайском языке // Национальная комиссия развития и реформ (НКРР) КНР // http://www.ndrc.gov.cn/zcfb/zcfbghwb/201604/t20160427_799898.html (дата обращения: 03.02.2019)

[КНР, 2017]План развития технологий искусственного интеллекта нового поколения 2017 // http://www.gov.cn/zhengce/content/2017-07/20/content_5211996.htm (дата обращения: 01.03.2019)

[Хант Э, 1978]. Искусственный интеллект: Пер. с англ. – М.: Мир, 1978. – 560 с. – С. 11

[Морхат, 2018] Морхат П.М. Право и искусственный интеллект: монография. Российская государственная академия интеллектуальной собственности. М.: ЮНИТИ-ДАНА, 2018, с.10, 20

[Calo, 2016] Ryan Calo, Robots in American Law https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2737598 (дата обращения 01.03.2019)

[BMVI, 2017] Ethics Commission Automated and Connected Driving, Federal Ministry of Transport and Digital Infrastructure https://www.bmvi.de/SharedDocs/EN/Documents/G/ethic-commission-report.pdf?__blob=publicationFile (дата обращения 01.03.2019)

Российская ассоциация искусственного интеллекта
Санкт-Петербургский институт информатики и автоматизации
Российской академии наук

ПЯТЫЙ ВСЕРОССИЙСКИЙ НАУЧНО-ПРАКТИЧЕСКИЙ СЕМИНАР «БЕСПИЛОТНЫЕ ТРАНСПОРТНЫЕ СРЕДСТВА С ЭЛЕМЕНТАМИ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА» (БТС-ИИ-2019)

22-24 мая 2019
г. Санкт-Петербург, Россия

Труды семинара

ISBN 978-5-6042802-0-1

Подписано в печать 24.04.2019
Тираж 100 экз. Формат 60x84/16
Печать – офсетная. Бумага офсетная. Заказ №5490.
Отпечатано в типографии «Политехника-сервис»
с оригинал-макета заказчика.