**Российская ассоциация искусственного интеллекта**


# ТРЕТИЙ ВСЕРОССИЙСКИЙ НАУЧНО-ПРАКТИЧЕСКИЙ СЕМИНАР «БЕСПИЛОТНЫЕ ТРАНСПОРТНЫЕ СРЕДСТВА С ЭЛЕМЕНТАМИ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА»

22-23 сентября 2015
г. Иннополис, Республика Татарстан, Россия

**Труды семинара**

УДК 004.8
ББК-32.813
Т 66

**Т 66 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. – 184 с.**

**ISBN 978-5-906895-89-9**

В сборник включены тексты работ, представленные на третьем Всероссийском научно-практическом семинаре «Беспилотные транспортные средства с элементами искусственного интеллекта» (БТС-ИИ-2016) 22-23 сентября 2016 года.

Официальный сайт семинара – www.ai-uv.ru

ISBN 978-5-906895-89-9

# 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 24 сентября 2014 года в Казани в рамках Четырнадцатой национальной конференции по искусственному интеллекту с международным участием (КИИ-2014). В 2015 году семинар прошел в Санкт-Петербурге в рамках Международной научно-технической конференции «ЭКСТРЕМАЛЬНАЯ РОБОТОТЕХНИКА». В этом году семинар БТС-ИИ – самостоятельное мероприятие, проводимое в новом инновационном Российском университете Иннополис. Программа семинара подразумевает выступление 2 приглашенных и 17 регулярных докладчиков, проведение круглого стола Фонда перспективных исследований, а также экскурсию по Иннополису.

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

**Программный комитет БТС-ИИ-2016**

## Организатор

Российская ассоциация искусственного интеллекта (www.raai.org)

## Организационная поддержка

Университет Иннополис (www.university.innopolis.ru)

## Программный комитет

**В.Е. Павловский (со-председатель)**, доктор физико-математических наук, профессор, главный научный сотрудник Института прикладной математики им. М.В. Келдыша РАН, член научного совета Российской ассоциации искусственного интеллекта.

**С.Б. Ткачев**, доктор физико-математических наук, профессор, лауреат премии правительства РФ в области науки и техники, профессор кафедры "Математическое моделирование" МГТУ им. Н.Э. Баумана.

**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 НИЦ "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.

**К.С. Яковлев (со-председатель)**, кандидат физико-математических наук, старший научный сотрудник Федерального исследовательского центра «Информатика и управление» Российской академии наук, член научного совета Российской ассоциации искусственного интеллекта.

**Е.А. Магид**, доктор технических наук, профессор кафедры Интеллектуальной Робототехники Высшей школы ИТИС, Казанский Федеральный Университет.

**И.М. Афанасьев**, кандидат технических наук, доцент Института Робототехники, Университет Иннополис.

## Официальный сайт семинара

www.ai-uv.ru

# Приглашенные доклады

# Интеллектуальные транспортные системы: прошлое, настоящее, будущее

Я.А. Холодов (*kholodov@crec.mipt.ru*)
Университет Иннополис, г. Иннополис
Московский физико-технический институт, г. Москва

В последние годы словосочетание «Интеллектуальные Транспортные Системы» (Intelligent Transport Systems) и соответствующие аббревиатуры – ИТС (ITS) – стали обычными в различных документах развитых стран.

«Интеллектуальные транспортные системы (ИТС) - это системная интеграция современных информационных и коммуникационных технологий и средств автоматизации с транспортной инфраструктурой, транспортными средствами и пользователями, ориентированная на повышение безопасности и эффективности транспортного процесса, комфортности для водителей и пользователей транспорта»

Использование ИТС в мире происходит в общественном транспорте, при повышении безопасности дорожного движения, ликвидации заторов на дорогах, повышения производительности транспортных систем.

Сегодня наиболее активно развиваются следующие технологии ИТС:

- Управление движением на автомагистралях;
- Предотвращение столкновений транспортных средств и безопасность их движения;
- Электронные системы оплаты транспортных услуг;
- Управление движением на городской уличной сети;
- Интермодальные грузовые перевозки;
- Контроль погоды на автодорогах;
- Управление общественным транспортом;
- Информационное обслуживание участников движения.

Одно из перспективных направлений развития ИТС – реализация концепции интеллектуального автомобиля. Существует международная программа «Транспортные средства повышенной безопасности». Первые результаты использования бортовых интеллектуальных систем показали, что они способны уменьшить число ДТП на 40%, а число ДТП со смертельным исходом на 50%. Развитие ИТС методологически базируется на системном подходе, формируя ИТС именно как системы, а не как отдельные модули (сервисы).

Концептуальную схему построения ИТС следует рассматривать как организацию системной формы взаимодействия всех видов транспорта и максимально эффективное использование транспортного ресурса за счет наиболее продвинутых вариантов поточных схем движения траффика, обеспечивающих качество транспортных услуг.

# Биологически инспирированные подходы в робототехнике

В.Э. Карпов (*karpov_ve@mail.ru*)
НИЦ «Курчатовский институт», г. Москва

В докладе рассматривается верификационный аспект биоинспирированности. Изучение особенностей поведения роботов, обладающих эмоциональной-темпераментной компонентой привело к выводу о том, что на этой базе вполне адекватно реализуются самые разнообразные поведенческие феномены, в частности – поведение зомбированного паразитами индивида, а также то, что в психиатрии называется раздвоением личности.

Особенность эмоциональной системы управления (СУ) робота заключается в наличии контуров положительной обратной связи, что делает ее весьма чувствительной к достаточно слабым воздействиям. Показано, каким образом паразит, будучи простым объектом с весьма ограниченным спектром возможностей, может коренным образом менять поведение робота (зомбировать его), воздействуя на потребностные и оценочные каналы СУ (аналог нейромодуляционного манипулирования), а также влияя на параметры контура обратной связи (непосредственное управление участками мозга).

Архитектура эмоциональной СУ отражает наличие потребностей и способов их удовлетворения (моторных программ). Это позволяет рассматривать СУ как совокупность агентов, обладающих своим целеполаганием, потребностями и способами их достижения. Это означает наличие в общем случае конфликтов между агентами, а эмоциональные обратные связи служат как раз средством согласования работы подсистем, стабилизации поведения системы в целом. Возникающие эффекты дисбаланса и автоколебаний в системе (фантомные сигналы и потребности) можно рассматривать с точки зрения психических расстройств.

Исследования рассмотренных феноменов паразитического зомбирования и шизофрении проводились на ряде имитационных моделях, в то время как архитектура эмоциональной и темпераментной систем управления отрабатывались на реальных робототехнических устройствах.

## Содержание

**УДК 519.878, 519.1, 004.942**

# ПЛАНИРОВАНИЕ МАРШРУТА ДЛЯ БЕСПИЛОТНОГО НАЗЕМНОГО РОБОТА С УЧЕТОМ МНОЖЕСТВА КРИТЕРИЕВ ОПТИМИЗАЦИИ

Р.О. Лавренов[a] (*r.lavrenov@innopolis.ru*)
И.М. Афанасьев[a] (*i.afanasyev@innopolis.ru*)
Е.А. Магид[b] (*dr.e.magid@ieee.org*)
[a] Университет Иннополис, г. Иннополис

[b] Казанский федеральный университет, г. Казань

**Аннотация**. Поиск оптимального маршрута движения для беспилотного транспортного средства является сложной задачей робототехники, требующей комплексного подхода. Основными критериями оценки качества алгоритма построения маршрута являются скорость выполнения поиска и оптимальность полученного для робота пути относительно различных, задаваемых пользователем, параметров оптимизации. В данной статье описан подход к планированию маршрута, позволяющий построить предварительный маршрут движения, а затем его динамически корректировать в режиме реального времени путем изменения весовых функций для различных параметров оптимизации. Мы представляем набор ключевых параметров оптимизации маршрута и результаты работы разработанного алгоритма в среде MATLAB.[1]

**Ключевые слова**: критерии оптимизации, диаграмма Вороного, планирование маршрута, беспилотный наземный робот (БНР)

## Введение

В настоящее время одной из наиболее важных задач робототехники является задача планирования пути для автономных роботов. Устройства при этом должны выполнять различные миссии: сбор информации в заранее неизвестных динамических средах, поисково-спасательные операции или решение сложных транспортно-логистических задач. При этом, при планировании маршрута для роботов всегда приходится учитывать множество факторов, зачастую выбирая между кратчайшей или наиболее безопасной траекторией движения робота. В данной статье мы

[1] Работа выполнена при финансовой поддержке РФФИ (проект № 15-57-06010).

рассматриваем набор ключевых параметров оптимизации пути робота и подход к планированию маршрута, согласно которому сперва, на стадии глобального планирования, строится предварительный маршрут движения, а затем, на стадии локального планирования, он динамически корректируется в режиме реального времени путем изменения весовых функций различных параметров оптимизации. Прототип алгоритма реализован в среде MATLAB, и результаты работы разработанного алгоритма представлены для различных целевых функций оптимизации.

# 1 Планирование маршрута в статической среде: Глобальные методы планирования

В начале исследований был проведен анализ существующих алгоритмов задачи планирования маршрута между двумя заданными точками (point-to-point navigation) на заранее известной статической карте. В задаче выбора пути для мобильного робота широко используется метод потенциального поля [Magid и др., 2006]: цель имеет некоторый положительный заряд, препятствия - отрицательный, местоположения цели и препятствий фиксированы. Для планирования маршрута робота моделируются действующие на него мнимые силы притяжения и отталкивания зарядов [Andrews и др., 1983], [Khatib O., 1986]. Метод потенциального поля продолжает использоваться и модифицироваться: так, в работе [Lei Tang и др., 2010] предложена модификация алгоритма по принципу гравитационной цепочки от начального к целевому положению.

Популярность приобрели методы, строящие граф вокруг препятствий на известной карте. Такой граф называется дорожной картой (Roadmap). Препятствия представляются в виде замкнутых 2D- или 3D-полигонов, и граф создается, используя вершины препятствий как узлы графа, которые соединяются ребрами графа, если препятствия не блокируют прямую линию между ними. Методы планирования пути, которые используют подобные графы, называются методами дорожной карты. Таковыми являются метод графа видимости (Visibility graph) [Simeon и др., 2000] и метод графа касательных (Tangent graph) [Liu и др., 1992]. Одним из простейших методов нахождения пути на вышеописанных графах является алгоритм A* (1968 г.), модернизирующий алгоритм Дейкстры.

Кроме того, для построения дорожной карты применяются методы клеточного разбиения. Два основных метода разбиения, при которых граф строится на линиях стыка ячеек, – это трапециевидная декомпозиция и декомпозиция Морзе. Для ускорения расчетов маршрута на больших картах, разработаны методы, основанные на сэмплировании, т.е. на снятии выборочных замеров с определенной степенью дискретизации и их сопоставления с маршрутами. Такими являются методы быстрорастущих

случайных деревьев (Rapidly-Exploring random trees, RRT) [LaValle S.M., 1998] и вероятностных дорожных карт [Kavraki и др., 1996]. В качестве примера, метод вероятностных дорожных карт использовался для навигации наземного робота через лабиринт в составе гетерогенной группы [Афанасьев и др., 2015].

В настоящее время для построения дорожной карты широко используется метод диаграммы Вороного. Для нашей задачи – это один из наиболее ценных методов, позволяющий строить максимально безопасный маршрут, так как ребра графа будут располагаться на наибольшем удалении от препятствий [Choset и др., 1997], [Lau и др., 2010]. Кроме того, найденные вершины графа можно использовать в алгоритме построения пути по сплайнам (Spline-based algorithm) [Magid и др., 2006], позволяющим вычислить наиболее гладкую траекторию движения робота, что также является важным критерием при оптимизации маршрута.

Анализ существующих решений для задачи планирования пути в MATLAB позволил выбрать программный инструмент RobotMotionToolbox [Gonzalez и др., 2015], в графическом интерфейсе которого можно создать карту с препятствиями в виде выпуклых полигонов. Среди функций данного программного средства реализованы такие алгоритмы, как Visibility graph, методы декомпозиции и др.

## 2 Планирование маршрута в динамической среде: Локальные методы планирования

В задачах поиска маршрута мобильным роботом важную роль играют и локальные методы планирования: в незнакомой или измененной среде робот должен уметь обнаруживать препятствия и динамически перестраивать свою траекторию для их обхода. Основными алгоритмами локального планирования пути являются методы семейства Bug, согласно которым робот движется по направлению к цели, огибая встреченные препятствия. Модификации алгоритмов, Bug1 и Bug2, различаются способом обхода препятствий, а метод DistBug [Kamon и др., 1997] использует для дистанционного обхода препятствий датчик расстояния. Алгоритм TangentBug расширяет DistBug создавая график локальных касательных (LTG) во время движения к цели с локальным выбором оптимального направления [Choset и др., 2005]. Его модификация CautionsBug [Magid и др., 2004] является более консервативным методом и строит LTG на каждом шаге, и превосходит TangentBug в общем случае.

Модификации метода потенциального поля для локального планирования пути основаны на гистограмме векторного поля (VFH). Этот алгоритм и основанные на нем методы VFH+ и VFH* используют

двумерную декартову сетку в качестве модели мира, в каждой ячейке которой определяется плотность препятствий [Ulrich и др., 1998]. При движении в динамических условиях часто применяют алгоритмы, базирующиеся на принципах вышеупомянутого алгоритма A*: Anytime Repairing A* (ARA*), Dynamic A* (D*), Lifelong Planning A* (LPA*) и наиболее популярные D* Lite [Likhachev и др., 2002] и Anytime D* (AD*).

В задачах динамической маршрутизации все чаще применяются методы глобального планирования и их модификации, так как даже в неизвестной среде, по мере построения карты, современные ЭВМ позволяют рассчитывать дорожную карту в реальном времени. В том числе, используется и алгоритм диаграммы Вороного, разбивающий пространство на сетку, каждая из ячеек которой может быть либо препятствием, либо пустой областью [Lau и др., 2013]. Однако, тестирование выявляет ограничения такого подхода, поскольку алгоритм сильно зависит от размера сетки и строит избыточную диаграмму Вороного с множеством лишних линий.

## 3 Поставленные задачи и настройки системы

Основной задачей нашего проекта является расчет в реальном времени маршрута движения беспилотного наземного робота (БНР) с учетом изменения критериев оптимизации маршрута (например, длина пути, кривизна и пр.). Исходным алгоритмом планировании пути выбран метод диаграммы Вороного, так как граф Вороного по определению представляет собой набор траекторий, наиболее безопасных для обхода препятствий, а при динамических изменениях первоначальный граф возможно частично пересчитать локально за ограниченное время. В рамках проекта используется российский БНР «Инженер» - гусеничный робот с дополнительными флипперами, способный преодолевать различные препятствия (Рис.1).

Рис. 1. Российский робот «Инженер», компания «Сервосила».

# 4 Выбор и реализация алгоритмов в MATLAB

## 4.1 Глобальный алгоритм маршрутизации

Реализация алгоритма планирования маршрута по диаграмме Вороного осуществлялась в среде MATLAB с использованием пакета RobotMotionToolbox. Алгоритм работает следующим образом: (1) От каждой грани препятствия с определенным шагом строятся лучи; (2) В местах пересечения лучей от разных граней ставятся точки; (3) Точки от соседних лучей соединяются в отрезки; (4) Соседние отрезки собираются в граф; (5) Конечный путь в графе находится по алгоритму Дейкстры.

Алгоритм был нами модифицирован под текущую задачу. Была добавлена возможность создавать круглые и невыпуклые препятствия, и метод построения диаграммы Вороного был соответственно изменен для работы со всеми типами препятствий. Были добавлены лучи от краев карты, тем самым обеспечивая построение диаграммы на карте с любым расположением препятствий и точек старта/цели. Пример такого расчета показан на Рис.2а. Для ускорения работы алгоритма, были убраны: точки пересечения лучей от невыпуклых препятствий и дублирование отрезков, что позволило находить точки ветвления графа, как места пересечения трех отрезков, а также его крайние точки, которые находились в углах карты (Рис.2б).

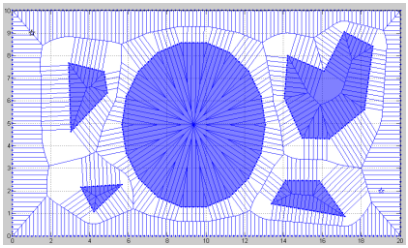


Рис. 2а. Предварительный расчет диаграммы Вороного для карты с препятствиями

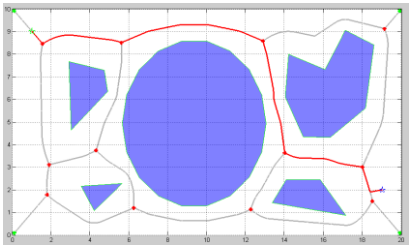


Рис. 2б. Финальная диаграмма Вороного с указанными точками ветвления графа и построенный по ней путь

Таким образом, была изменена реализация алгоритма по диаграмме Вороного, чтобы он работал не только с точками, как в работе [Lau и др., 2013], но и строил граф при любых положениях препятствий, начальной и конечной точек маршрута. Непосредственно ребра графа (упорядоченные группы отрезков от вершины к вершине) стали теми кривыми, учитывая которые при расчете, можно рассчитать различные гомотопии маршрутов

на карте. По определению гомотопии, два маршрута принадлежат к одному гомотопическому классу, если возможно преобразование одного маршрута в другой при помощи неразрывной деформации.

## 4.2 Параметры оптимизации маршрута

Построенный маршрут должен удовлетворять определенным требованиям целевой функции. При этом маршрут должен каждый раз динамически пересчитываться при изменении параметров этой функции, т.е. критериев оптимизации маршрута. Изучая методы оптимизации рассчитанного маршрута, были выделены следующие ключевые параметры целевой функции:

- ***Время прохождения*** **–** определяет насколько важно пройти маршрут самым быстрым образом. Значение коррелирует с длиной и сглаженностью маршрута, и другими критериями.
- ***Длина пути*** – характеризует, насколько важно пройти кратчайшим путем в рассчитанной гомотопии.
- ***Максимальное (****или среднее****) расстояние до препятствий по мере прохождения маршрута*** – определяет насколько важно роботу во время движения сохранять дистанцию до препятствий.
- ***Кривизна пути*** (производная) – выявляет насколько важно, чтобы при движении по маршруту не было резких изменений траектории [Magid и др., 2006].
- ***Время прямой видимости начальной позиции и время выхода на прямую до целевой позиции*** – нетривиальный критерий, оценивающий важность быть на прямой видимости со стартовой или финишной точкой. Данный параметр нужен в случае, когда, например, в начальной точке у робота есть связь с устройством контроля, помогающим в планировании пути, и требуется выбрать маршрут, как можно дольше поддерживающий эту связь.

Кроме того, если на карте расположить критические точки опасности или, наоборот, роутеры связи, которые надо держать на определенном расстоянии, то возникают еще несколько параметров:

- ***Максимальное допустимое расстояние до роутеров связи*** – чтобы не терять контакт с контролирующим устройством и при этом обследовать карту или следовать выбранному маршруту.
- ***Длина пути, укрытая от опасностей***, – например, если в «точке опасности» находится источник радиации или открытый огонь, то следует выбирать маршрут так, чтобы при его прохождении робот был максимально загорожен препятствиями со стороны опасностей.
- ***Минимальное допустимое расстояние от точек опасности*** –

учитывает безопасную дистанцию во время движения робота.

## 4.3 Результаты симуляции в среде MATLAB

В настоящее время в нашей аппликации реализованы следующие параметры: длина пути, расстояние от препятствий, кривизна, время выхода на целевую позицию, а также связанные с критическими точками параметры. Например, для нахождения кратчайшего пути в пределах, удовлетворяемых критериям точек опасности, использовался алгоритм поиска пути по графу видимости. Если увеличивать значение критерия максимального расстояния от препятствий, то выбирается путь, рассчитанный по диаграмме Вороного (Рис.3а-г). Заменой ребер диаграммы Вороного в первоначальном маршруте создаются маршруты в различных гомотопиях, предоставляя пути для параметра времени прямой видимости точек старта и цели. Для этого также сравниваются расстояния от точек старта/цели до первой кривой диаграммы (Рис.4а,б). Аналогично, замена вершин графа видимости позволяет вычислять различные гомотопии в графе. Далее, находятся соответствия между гомотопиями двух графов. Таким образом, для каждого пути из диаграммы Вороного будет находится кратчайший путь из графа видимости.

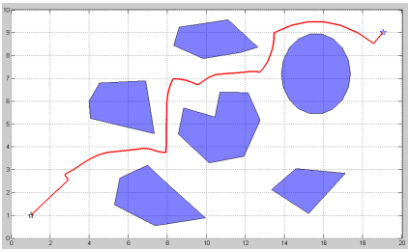


Рис. 3а. Критерий расстояния до препятствий – 100%, длины пути 0%

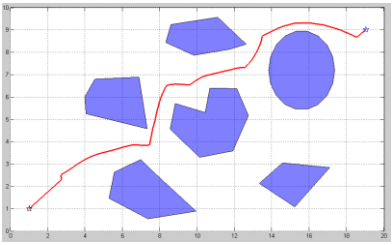


Рис. 3б. Критерий расстояния до препятствий – 65%, длины пути 35%

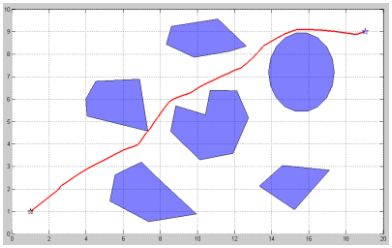


Рис. 3в. Критерий расстояния до препятствий – 35%, длины пути 65%

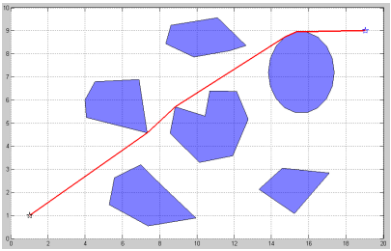


Рис. 3г. Критерий расстояния до препятствий – 0%, длины пути 100%

Соответственно, если воспользоваться методом планирования по сплайнам [Magid и др., 2006] и соединять сплайнами вершины диаграммы Вороного (красные точки на Рис.2), то путь будет удовлетворять критерию минимальной кривизны (производной) траектории пути.

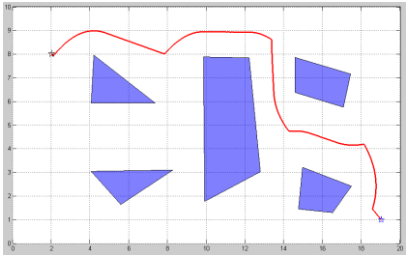


Рис. 4а. Критерий расстояния – 100%, время прямой видимости - 0%

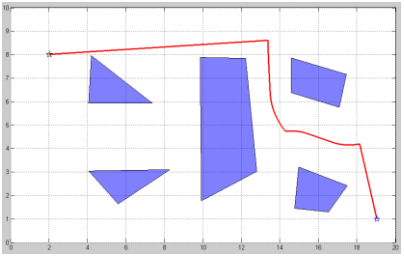


Рис. 4б. Критерий расстояния –0%, время прямой видимости - 100%

При расчетах сначала рассчитываются траектории в различных гомотопиях, удовлетворяющие 100% значениям критериев. Затем, в зависимости от процентного вклада каждого критерия, вычисляется путь.

Если на карте отметить критические точки опасности (или роутеры связи), то процесс планирования пути будет аналогичным. Среди гомотопий будет выбираться та, которая будет удовлетворять критериям удаленности от точек опасности (или доступности роутеров связи) и в дальнейшем к данной гомотопии будут применяться критерии для обычного пути, описанные выше (Рис.5).

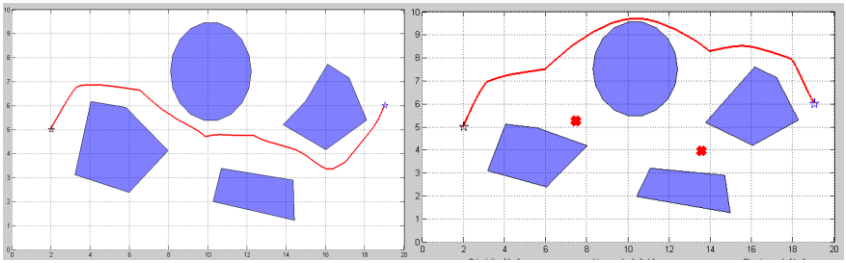


Рис. 5. Примеры расчета траекторий при критерии дистанции от опасностей 100% и критерии длины 50% на карте без опасностей (слева) и с опасностями (справа)

При появлении на карте критических точек опасности, алгоритмы учитывают их как блокирующие маршрут препятствия, через которые не будут проходить пути, как бы много их не было на карте. Расчет диаграммы Вороного на карте с критическими точками показан на Рис.6.

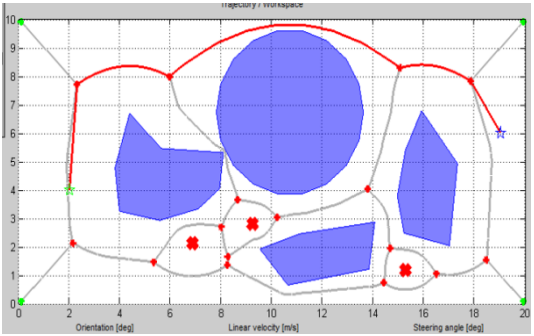



Рис. 6. Расчет пути по диаграмме Вороного с учётом критических точек

## 5 План дальнейшей работы

В качестве следующих шагов предполагается интегрировать в алгоритм дополнительные параметры целевых функций, в том числе, связанные с неровностями поверхности, что потребует применение алгоритмов локального планирования пути. Кроме того, будет проведен анализ корреляции между связанными параметрами оптимизации пути и настройка работы планирования маршрута в среде с динамическими препятствиями. В дальнейшем, алгоритмы предполагаются к реализации на языке C++ под робототехническую операционную систему ROS с применением как в 3D симуляторе, так и на реальном гусеничном роботе «Инженер».

## Заключение

Важным результатом данной работы является создание алгоритма планирования пути, учитывающего изменяемые в реальном времени параметры оптимизации. Фактически, метод интегрирует известные подходы к глобальному и локальному планированию маршрутов, оптимизируя рассчитываемые траектории мобильного робота в любой, подходящей по критериям оптимизации, гомотопии. Таким образом, сперва на стадии глобального планирования строится предварительный маршрут движения, а затем на стадии локального планирования он динамически корректируется в режиме реального времени путем изменения весовых функций различных параметров оптимизации. Разработанный алгоритм демонстрирует быструю и робастную подстройку к динамически меняющейся среде, обеспечивая эффективную маршрутизацию мобильных роботов.

## Список литературы

**[Афанасьев и др., 2015]** И.М. Афанасьев, А.Г. Сагитов, И.Ю. Данилов, Е.А. Магид. Навигация гетерогенной группы роботов (БПЛА и БНР) через лабиринт в 3D симуляторе Gazebo методом вероятностной дорожной карты // Труды семинара БТС-ИИ-2015, Санкт-Петербург: «Политехника-сервис», 2015.

**[Andrews и др., 1983]** Andrews J. R. and Hogan N. Inpedance Control as a Framework for Implementing Obstacle Avoidance in a Manipulator // Control of Manufacturing Processes and Robotic Systems, 1983.

**[Choset и др., 1997]** Choset H. and Burdick J. Sensor Based Motion Planning: The Hierarchical Generalized Voronoi Graph // Advanced Robotics, 1997.

**[Choset и др., 2005]** Choset H., Lynch K., Hutchinson S., Kantor G., Burgard W., Kavraki L. and Thrun S. Principles of Robot Motion: Theory, Algorithms, and Implementations // The MIT Press, 2005.

**[Gonzalez и др., 2015]** Gonzalez R., Mahulea C. and Kloetzer M. A Matlab-based Interactive Simulator for Teaching Mobile Robotics // IEEE CASE'2015: Int. Conf. on Autom. Science and Engineering, 2015

**[Kamon и др., 1997]** Kamon I. and Rivlin E. A new range-sensor based globally convergent navigation algorithm for mobile robots // The International Journal of Robotics Research, 1997

**[Kavraki и др., 1996]** Kavraki L. E., Svestka P., Latombe J.-C. and Overmars M. H., Probabilistic roadmaps for path planning in high-dimensional configuration spaces // Transactions on Robotics and Automation, 1996

**[Khatib O., 1986]** Khatib O. Real-Time Obstacle Avoidance for Manipulators and Mobile Robots // The International Journal of Robotics Research, 1986

**[Lau и др., 2010]** Lau B., Sprunk C. and Burgard W. Improved Updating of Euclidean Distance Maps and Voronoi Diagrams // Intelligent Robots and Systems, 2010

**[Lau и др., 2013]** Lau B., Sprunk C. and Burgard W. Efficient Grid-Based Spatial Representations for Robot Navigation in Dynamic Environment // Robotics and

Autonomous Systems, 2013

**[LaValle S.M., 1998]** LaValle S.M. Rapidly-exploring random trees: A new tool for path planning // Computer Science Department, Iowa State University, 1998

**[Lei Tang и др., 2010]** Lei Tang, Songyi Dian and Gangxu Gu and Kunli Zhou A novel potential field method for obstacle avoidance and path planning of mobile robot // Computer Science and Information Technology, 2010

**[Likhachev и др., 2002]** Likhachev M. and Koenig S. D* lite // Eighteenth national conference on Artificial intelligence, 2002

**[Liu и др., 1992]** Liu Y. and Arimoto S. Path planning using a tangent graph for mobile robots among polygonal and curved obstacles // Int. J. of Robotics Research, 1992

**[Magid и др., 2004]** Magid E., and Rivlin E. CautiousBug: a competitive algorithm for sensory-based robot navigation // Intelligent Robots and Systems, 2004

**[Magid и др., 2006]** Magid E., Keren D., Rivlin E. and Yavneh I. Spline-Based Robot Navigation // Intelligent Robots and Systems, 2006

**[Simeon и др., 2000]** Simeon T., Laumond J.-P. and Nissoux C. Visibility based probabilistic roadmaps for motion planning // Advanced Robotics, 2000

**[Ulrich и др., 1998]** Ulrich I. and Borenstein J. VFH+: Reliable Obstacle Avoidance for Fast Mobile Robots // Robotics and Automation, 1998

**УДК 004.896**

# ROS-МОДЕЛИРОВАНИЕ ВЗАИМОДЕЙСТВИЯ БПЛА И НАЗЕМНОГО БЕСПИЛОТНОГО РОБОТА ДЛЯ РЕШЕНИЯ ЗАДАЧИ ПЛАНИРОВАНИЯ МАРШРУТА В СТАТИЧЕСКОЙ СРЕДЕ

А.Р. Габдуллин[a] (*a.gabdullin@innopolis.ru*)
А.К. Буйвал[a] (*alexbuyval@gmail.com*)
Р.О. Лавренов[ab] (*r.lavrenov@innopolis.ru*)
Е.А. Магид[b] (*dr.e.magid@ieee.org*)
[a] Университет Иннополис, г. Иннополис

[b] Казанский Федеральный Университет, г. Казань

**Аннотация**. Для эффективной навигации беспилотного наземного робота (БНР) необходимо иметь полноценную карту местности, которая во многих случаях недоступна и по разным причинам не может быть построена самим БНР. Задача картографирования может быть облегчена путем привлечения беспилотных летательных аппаратов (БПЛА), действующих совместно с БНР. В статье описываются особенности программного решения для кооперации БНА с несколькими БПЛА для задачи совместного построения трехмерных карт, получаемых с воздуха, и дальнейшее построение маршрута БНР с использованием полученной карты. Программная реализация алгоритмов осуществлена в среде ROS с использованием симулятора Gazebo. [1]

**Ключевые слова**: SLAM, планирование пути, автономный робот, беспилотный наземный робот (БНР), ROS, Gazebo, octomap, диаграмма Вороного

## Введение

Задача координации действий группы автономных роботов признается одной из ключевых при построении мультиагентных систем, и один из частных примеров - задача совместного картографирования.

Построение карты является необходимым первоначальным этапом перед планированием маршрута беспилотного наземного робота (БНР), так как именно карта представляет собой необходимое входное условие

1 Работа выполнена при финансовой поддержке РФФИ (проект № 15-57-06010).

для работы алгоритмов планирования пути. Однако, в реальных условиях, картографирование с БНР не всегда возможно в силу ограниченной его проходимости. С другой стороны, использование беспилотных летательных аппаратов (БПЛА) позволяет преодолеть это ограничение. Таким образом, группа из нескольких БПЛА и БНР может решать задачи построения маршрута на пересеченной местности, в том числе и в динамически изменяемых условиях.

В данной статье описывается способ передачи и объединение данных об окружающей среде с нескольких БПЛА напрямую для использования наземным аппаратом, а также предлагается практическое решение для планирования пути в статических условиях**.**

В рамках решения текущей задачи группового взаимодействия нами используется программная модель БНР «Husky» компании Clearpath (Рис.1а) и БПЛА мультироторного типа из библиотеки «hector_quadrotor» (Рис.1б) для фреймворка ROS (Robot Operating System).

Рис. 1а. Мобильный робот Clearpath Husky

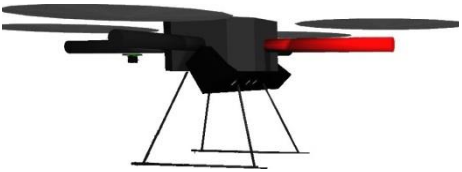


Рис. 1б. Модель квадрокоптера в Gazebo

## 1 Планирование маршрута в статической среде: Глобальные методы планирования

Методы планирования пути для БНР могут быть условно разделены на два типа: глобальный и локальный. Глобальный путь указывает БНР маршрут без учёта механических особенностей БНР, свойств поверхности движения и внезапных динамических изменений окружающей среды.

Среди глобальных методов планирования [Latombe, 2012] широко известны такие как:

- граф видимости [Latombe, 2012]
- диаграмма Вороного [Choset и др., 1997]
- метод потенциального поля [Magid и др., 2006]
- алгоритмы RRT, методы быстрорастущих случайных деревьев (англ. rapidly exploring random trees) [Melchior и Simmons, 2007]
- клеточное разбиение (англ. cell decomposition) [Lingelbach, 2004]

Для наших исследований основным методом планирования был выбран метод динамической диаграммы Вороного [Lau и др., 2010] ввиду его достаточной безопасности с точки зрения возможности столкновения с препятствиями и относительно невысокой вычислительной сложности.

## 2 Планирование маршрута в статической среде: Локальные методы планирования

Задача локального планирования [Choset, 2005] заключается в поиске пути с учётом возникающих динамических препятствий, погрешностей одометрии и информации о локально-окружающей среде. Примером алгоритма локальной навигации является семейство алгоритмов класса Bug [Magid и др., 2004]. Однако, эти алгоритмы полностью абстрагируются от физических свойств реального робота и используют исключительно локальные данные с карты, и соответственно, являются сильно теоретизированными и не оптимальными в условиях наличия даже частичной карты местности. Поэтому в качестве метода локальной планировки маршрута нами используется метод динамического окна (Dynamic Window Approach) [Fox, 1997]; основное его преимущество заключается в учете динамики робота, что даёт возможность совершать безопасные манёвры, не сталкиваясь с препятствиями.

## 3 Карты

Применяемые нами алгоритмы построения маршрута используют в качестве входной информации так называемые решётки занятости (англ. occupancy grid) [Elfes, 1989]. Их можно представить в виде матрицы целых чисел, представляющих собой степень проходимости ячейки пространства (0 – ячейка абсолютно непроходима, 255 – полностью проходима); таким образом, карта может быть представлена в виде растрового изображения в оттенках серого.

Однако, решётки занятости являются двумерными структурами, а сенсоры БПЛА возвращают трехмерное облако точек. Задача навигации в трехмерном пространстве существенно сложнее в решении, а также хранение и обработка облаков точек требует значительных вычислительных ресурсов. Поэтому, было решено совместить трехмерные

и двумерные подходы. В частности, чтобы сократить объем потребляемой облаком точек памяти, применялся подход октокарт [Hornung и др., 2013], основанный на использовании октодеревьев. Каждый узел дерева описывает степень проходимости воксела (дискретного элемента объема пространства). Узел, в свою очередь, может быть подразбит на восемь равных вокселов и т.д. (Рис.2а). Такая структура требует хранения лишь степени занятости воксела, но не его координат. Координаты могут быть вычислены динамически зная положение воксела в дереве. Также, в таком подходе, возможен выбор степени детализации октокарты. Для этого необходимо лишь ограничить глубину спуска по дереву. Это позволяет снизить вычислительную нагрузку на менее ответственных или однородных участках маршрута. Пример октокарты изображен на Рис.2б.

Проецируя вокселы на плоскость движения БНР, можно построить решетку занятости. Тем самым появляется возможность применения двумерного планирования маршрута. В нашем случае, каждый элемент решетки заполняется абсолютной максимальной высотой воксела в текущей точке пространства (т.е. максимальным значением по оси Z воксела со степенью занятости более 0 при текущих координатах (x,y) ).

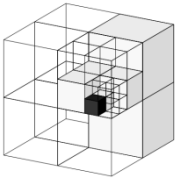


Рис. 2а. Восьмеричное разбиение, используемое в октокартах **[**Hornung и др., **2013]**

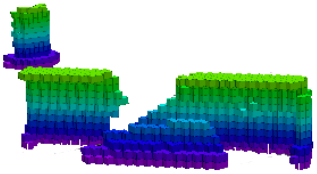


Рис. 2б. Два препятствия БНР в трехмерном пространстве, изображеные вокселами.

## 4 Поставленные задачи и настройки системы

Основная задача, которая ставилась в данном исследовании, – моделирование совместного исследования пространства и построения карты при помощи БПЛА с дальнейшим построением маршрута для БНР. В качестве БНР использовался робот Husky от компании Clearpath. Выбор данной модели был обусловлен хорошей документированностью программных средств, а также простотой модели управления.

Программной средой выбран фреймворк ROS Indigo. ROS являет собой набор программных модулей и интерфейсов, но не дает возможности симулировать физические явления. Такой возможностью

обладает симулятор Gazebo 2.2, который использовался нами в связке с ROS.

Список основный пакетов ROS, использованных для моделирования и симуляции:

- **husky_simulator** – симулятор наземного робота Husky
- **husky_navigation** – стек навигации наземного робота (глобальное и локальное планирование маршрута)
- **hector_quadrotor** – симулятор квадрокоптера с установлен-ным на него RGBD-сенсором (Kinect)
- **gmapper** – картографирование мобильным роботом
- **octomap** – трёхмерное картографирование и проецирование двумерной карты
- **map_server** – хранение, сохранение и загрузка двумерных карт в виде решёток занятости
- **tf** – работа с трансформациями между различными системами координат
- **voronoi_planner** – глобальный планировщик пути по диаграм-ме Вороного
- **rviz** – средство визуализации

Пакет octomap, как следует из названия, отвечает за построение, обработку и хранение октокарт. Выбор пакета обусловлен эффективностью структуры хранения данных по потреблению памяти, возможностью изменять динамически уровень детализации карты. Данное преимущество является важным, т.к. объем полученных с сенсоров летальных аппаратов данных велик и, соответственно, хранение и обработка их в виде облака точек ресурсоемко. Также, пакет octomap строит решетки занятости по октокартам, что требуется для планирования маршрута. Для верного построения октокарты необходим программный узел объединения облаков точек, полученных с сенсоров Kinect с различных квадрокоптеров. Это возможно лишь при условии наличия верных трансформаций между системами координат Kinect и картой БНР соответственно.

# 5 Моделирование «Husky» и квадрокоптера в ROS

Для моделирования квадрокоптеров использовался программный модуль (т. н. пакет библиотеки) фреймворка ROS *hector_quadrotor*. В качестве основного сенсора картографирования было принято решение использовать цвето-глубинометр (RGBD-сенсор) Kinect. Этот выбор был обусловлен планами по дальнейшему развитию проекта и доступным оборудованием. Также возможно использование стереокамер. Как БПЛА, так и БНР, используют Unified Robot Description Format (URDF) для

описания физических и механических свойств модели. Однако, в комплекте поставки пакета *hector_quadrotor* сенсор Kinect установлен в плоскости, параллельной плоскости вращения несущих винтов, то есть с нулевым тангажным углом. Учитывая то, что крен БПЛА во время следования маршруту невелик, сенсор захватывает поверхность местности недостаточно. Для решения данной проблемы, при моделировании Kinect был повернут нами на 45 градусов вниз (Рис.1б). Пример совместного расположения группы роботов, состоящей из одного БНР и двух БПЛА-квадрокоптеров, в смоделированном мире с простыми препятствиями показан на Рис.3.

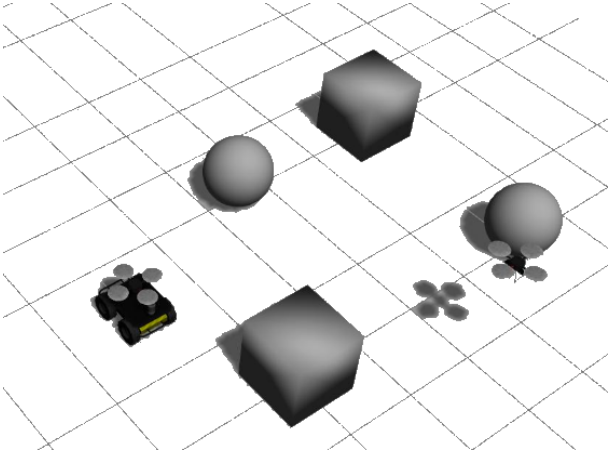


Рис. 3. Симуляция совместной работы БНР и 2-х БПЛА в Gazebo 2.2

## 6 Совместная работа «Husky» и квадрокоптеров в ROS

Немаловажным моментом в движении группы роботов является задача координации взаимного местоположения роботов. В целях избегания столкновений, все БПЛА были расположены на заведомо безопасном удалении друг от друга, позволяющим перекрывать области видимости сенсоров Kinect друг друга. Для следования была выбрана траектория, подобная меандру, также известная как зигзаг-бустрофедон [Galceran и Carreras, 2013]. Аналогичные траектории используются, например, при подводной съемке [Багницкий и др, 2010]. Пример октокарты, построенной таким образом, изображен на Рис.4а, ее проекция представлена на Рис.4б.

Ключевым фактором успешного построения октокарты является точная локализация каждого робота и передача правильной матрицы

трансформации в общее дерево трансформаций (пакет *tf*). На текущем этапе локализация БПЛА использовала информацию симулятора Gazebo об истинном положении аппарата (так называемая ground truth локализация). Этот подход в дальнейшем будет заменён на локализацию по сенсорам БПЛА.

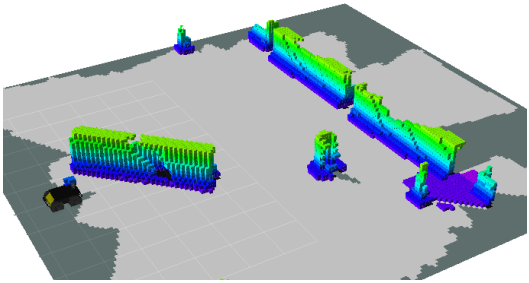


Рис. 4а. Октокарта, построенная двумя БПЛА, следующими по меандру вперед от места стоянки БНР (БНР расположен на левом крае карты, по центру)

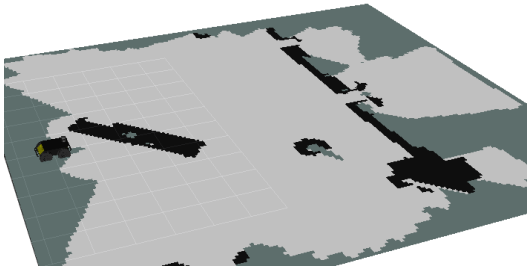


Рис. 4б. Двумерная проекция карты с Рис.4а (решётка занятости)

Для решения задачи планирования маршрута использовался алгоритм динамической диаграммы Вороного [Lau и др., 2013]. Он позволяет использовать дискретное пространство (решетку занятости), и при этом минимизирует количество ячеек, требующих обновления, в случае возникновения динамических препятствий. На Рис.5 показана построенная диаграмма при максимальном разрешении воксела в 1 кубический дециметр.

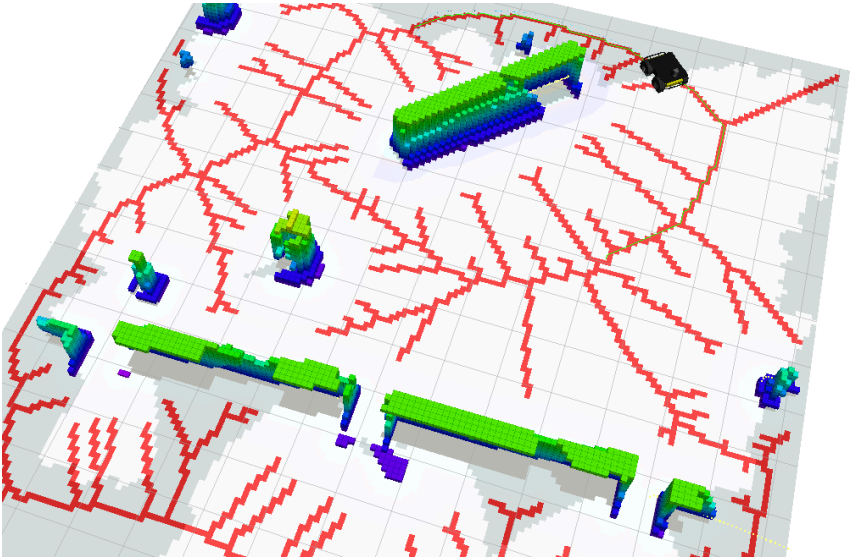


Рис. 5. Робот Husky (правый верхний угол) на карте, построенной двумя БПЛА

Разрешение карты важно для планировщика маршрута *voronoi_planner*, так как при высоком разрешении существенно увеличивается время, необходимое на расчет диаграммы Вороного и построение маршрута движения уже с использованием ребер и узлов диаграммы. Однако, при разрешении карты в 1 куб. дециметр, планировщик справляется со своей задачей за время менее 1 секунды для площадей порядка сотен квадратных метров. Пример глобального маршрута движения БНР показан на Рис.6 черной линией.

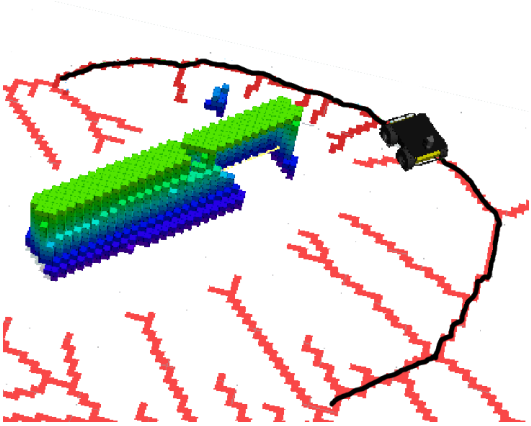


Рис. 6. БНР следует по глобальному маршруту (показан черным цветом), построенному по диаграмме Вороного (увеличенный масштаб карты с Рис.5)

## 6 План дальнейшей работы

На следующем шаге наших исследований предполагается внедрить более реалистичную модель БПЛА, использующую методы локализации, основывающиеся на данных с датчиков оптического потока, и применение алгоритмов визуальной одометрии. Также, мы планируем модифицировать программный комплекс для возможности работы в распределенной компьютерной системе, так как подобное ROS-моделирование требует значительных вычислительных ресурсов.

## Заключение

В статье описывается программное решение для кооперации БНР с несколькими БПЛА для построения карт и поиска маршрута движения БНР с использованием полученной карты. Программная реализация алгоритмов осуществлена в среде ROS с использованием симулятора Gazebo. Нами был создан симулятор, позволяющий моделировать различные условия местности и осуществлять её картографирование с воздуха при помощи группы БПЛА. Поиск маршрута движения для БНР осуществлен на полученной карте при помощи диаграммы Вороного. Симулятор также позволяет тестировать различные алгоритмы планирования маршрута в статической среде. В дальнейшем, помимо усовершенствования модели БПЛА и использования распределенной компьютерной системы, нами планируется расширить алгоритмы картографирования и поиска маршрута для возможности их использования в динамической среде.

## Список литературы

**[Buniyamin, 2011]** Buniyamin N. et al. A simple local path planning algorithm for autonomous mobile robots //International journal of systems applications, Engineering & development. – 2011. – Т. 5. – №. 2. – С. 151-159.

**[Choset и др., 1997]** Choset H. and Burdick J. Sensor Based Motion Planning: The Hierarchical Generalized Voronoi Graph // Advanced Robotics, 1997.

**[Choset, 2005]** Choset H. M. Principles of robot motion: theory, algorithms, and implementation. – MIT press, 2005.

**[Elfes, 1989]** Elfes A. Using occupancy grids for mobile robot perception and navigation //Computer. – 1989. – Т. 22. – №. 6. – С. 46-57.

**[Fox, 1997]** Fox D., Burgard W., Thrun S. The dynamic window approach to collision avoidance //IEEE Robotics & Automation Magazine. – 1997. – Т. 4. – №. 1. – С. 23-33.

**[Gonzalez и др., 2015]** Gonzalez R., Mahulea C. and Kloetzer M. A Matlab-based Interactive Simulator for Teaching Mobile Robotics // IEEE CASE'2015: Int. Conf. on Autom. Science and Engineering, 2015.

**[Galceran и Carreras, 2013]** Galceran, E., and Carreras, M. A survey on coverage path planning for robotics // Robotics and Autonomous Systems 61, no. 12, C.1258-1276, 2013.

**[Hornung и др., 2013]** Armin Hornung, Kai M. Wurm, Maren Bennewitz, Cyrill Stachniss and Wolfram Burgard, OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees //Autonomous Robots, 2013.

**[Latombe, 2012]** Latombe J. C. Robot motion planning. – Springer Science & Business Media, 2012. – Т. 124.

**[Lau и др., 2010]** Lau B., Sprunk C. and Burgard W. Improved Updating of Euclidean Distance Maps and Voronoi Diagrams // Intelligent Robots and Systems (IROS), 2010.

**[Lau и др., 2013]** Lau B., Sprunk C. and Burgard W. Efficient Grid-Based Spatial Representations for Robot Navigation in Dynamic Environment // Robotics and Autonomous Systems, 2013.

**[Lingelbach, 2004]** Lingelbach, F. Path planning using probabilistic cell decomposition // IEEE ICRA, Vol. 1, C. 467-472, 2004.

**[Magid и др., 2004]** Magid E., and Rivlin E. CautiousBug: a competitive algorithm for sensory-based robot navigation // IEEE IROS, 2004.

**[Magid и др., 2006]** Magid E., Keren D., Rivlin E. and Yavneh I. Spline-Based Robot Navigation // IEEE IROS, 2006.

**[Melchior и Simmons, 2007]** Melchior, Nik A., and Reid Simmons. Particle RRT for path planning with uncertainty // IEEE Robotics and Automation, С. 1617-1624, 2007.

**[Багницкий и др, 2010]** Багницкий А. В., Инзарцев А. В. Автоматизация подготовки миссии для автономного необитаемого подводного аппарата в задачах обследования акваторий //Подводные исследования и робототехника. – 2010. – №. 2. – С. 10.

**УДК 004.8**

# МЕТОД РАЗРЕШЕНИЯ КОНФЛИКТОВ ПРИ ПЛАНИРОВАНИИ ПРОСТРАНСТВЕННЫХ ТРАЕКТОРИЙ ДЛЯ ГРУППЫ БЕСПИЛОТНЫХ ЛЕТАТЕЛЬНЫХ АППАРАТОВ

А.А. Андрейчук (*andreychuk@mail.com*)
РУДН, Москва

К.С. Яковлев (*yakovlev@isa.ru*)
ФИЦ ИУ РАН, Москва

**Аннотация**. В работе рассматривается задача планирования совокупности траекторий для группы интеллектуальных агентов (беспилотных летательных аппаратов) в двухмерном случае. Исследуется децентрализованный подход к ее решению, когда процесс построения траекторий осуществляется независимо, а согласование и устранение конфликтов – централизовано. Предлагается новый метод разрешения конфликтов, использующий как механизмы задержки агентов, так и оригинальную процедуру локального перепланирования траектории.[1]

**Ключевые слова**: планирование траектории, мультиагентное планирование, разрешение конфликтов, беспилотный летательный аппарат, мультиагентные системы.

## Введение

Задача планирования траектории в искусственном интеллекте и робототехнике обычно рассматривается как задача поиска пути на графе, вершинам которого соответствуют положения агента в пространстве, а ребрам – возможные переходы между ними (элементарные траектории). При этом задача планирования для коалиции агентов является NP-сложной [Yu, 2013]. Получение оптимальных решений этой задачи при достаточно большом числе агентов и размере графа весьма затруднительно, однако именно задачи с такими характеристиками обычно встречаются на практике, например, в области автоматизации управления коалициями беспилотных транспортных средств. Поэтому весьма распространен децентрализованный подход к решению задачи

[1] Работа выполнена при финансовой поддержке РФФИ (проект № 15-37-20893).

планирования, когда каждый агент строит свою траекторию независимо, а затем эти траектории централизованно согласуются между собой. Алгоритмы, реализующие этот подход, такие как, например, MAPP [Wang, 2011] или WHCA* [Silver, 2005], характеризуются высокой скоростью работы и активно применяются на практике. При этом эти алгоритмы способны функционировать лишь в условиях, когда конфликт между индивидуальными решениями (траекториями) возникает в определенной вершине графа. Однако известно, что в области планирования траектории для беспилотных транспортных средств, предпочтительнее использовать методы планирования, которые позволяют строить траектории, лишь частично привязанные к топологии графа. А именно – вершины графа используются в качестве опорных точек траектории, но сегменты траектории (ребра графа) генерируются на лету. К таким алгоритмам относятся, например, Theta* [Nash, 2007], Anya [Harabor, 2013], LIAN [Yakovlev, 2015]. Известные алгоритмы разрешения конфликтов не подходят для обработки траекторий, построенных этими алгоритмами, т.к. конфликты могут возникать в областях пространства, не соответствующих вершинам графовой модели окружающей среды. Таким образом, актуальной является задача разработки новых методов и алгоритмов планирования траектории для совокупности беспилотных транспортных средств, лишенных указанных недостатков. Именно эта задача решается в данной работе.

## 1 Модельная задача

Рассматривается задача автоматизации маловысотного полета группы малых беспилотных летательных аппаратов (в терминологии искусственного интеллекта – агентов) в городских условиях. Исследуется следующий сценарий: $n$ идентичных по своим характеристикам БЛА находятся на земле, затем набирают определенную высоту и осуществляют полет в горизонтальной плоскости (облетая препятствия) до пункта назначения, в котором совершают посадку. Скорость перемещения всех БЛА полагается одинаковой. Известны траектории отдельных БЛА, которые могут быть конфликтными в совокупности, т.е. два (или более БЛА) могут оказаться в одной точке пространства одновременно. Требуется разрешить все конфликты и получить такие траектории, следование по которым позволит избежать столкновений, как с препятствиями, так и друг с другом.

## 1 Формальная постановка задачи

Дан граф специального вида, МТ-Граф [Яковлев, 2013], который представлен в виде матрицы $A_{H \times W}=\{a_{ij}\}$, где $i, j$ – индексы вершин (клеток)

графа, а $H$, $W$ – его размеры. Даны $n$ путей на этом графе: $\pi^{(1)}$, … $\pi^{(n)}$. Путь $\pi$ – это последовательность секций $\pi=\{e_1, \ldots, e_v\}$, а секция $e=\langle a_{ij}, a_{kl}\rangle=\langle sp(e), ep(e)\rangle$, это проходимая прямая линия соединяющая центры соответствующих клеток – см. рис. 1.

Длиной секции, $len(e)$, будем называть расстояние между её концами. Секцию, длина которой после округления равна некоторому целому $\Delta$, будем называть $\Delta$-секцией. Длиной пути $\pi$ является сумма длин всех секций, составляющих этот путь: $len(\pi)=len(e_1) + \ldots + len(e_v)$.

Здесь и далее будем рассматривать только пути, состоящие из $\Delta$-секций. Построение таких путей может быть осуществлено, например, алгоритмом LIAN [Yakovlev et al., 2015] (который помимо прочего учитывает ограничение на угол отклонения между секциями) или незначительно модифицированным алгоритмом Theta* [Nash et al., 2007].

Потенциальным решением назовем набор частичных потенциальных решений (partial potential solutions) $PS=\{PPS^{(1)}, \ldots, PPS^{(n)}\}$, где $PPS^{(i)}$ это пара $\langle\pi^{(i)}, t^{(i)}\rangle$, $\pi^{(i)}$ –i-й путь, $t^{(i)}$ – i-я задержка.

Будем считать, что за 1 единицу времени агент преодолевает расстояние равное расстоянию между центрами двух горизонтально смежных вершин. В этих же единицах будем измерять величину $t^{(i)}$. Таким образом общая временная стоимость решения равна сумме стоимостей каждого частичного решения: $cost(PS) = cost(PPS^{(1)}) + \ldots + cost(PPS^{(n)})$, $cost(PPS^{(i)})=t^{(i)}+len(\pi^{(i)})$.

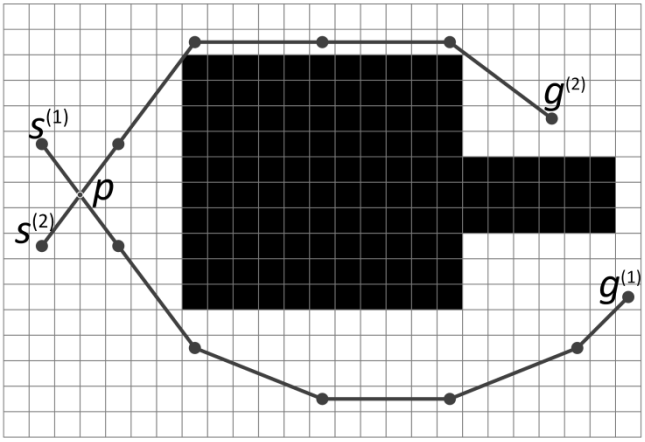



Рис.1 Примеры путей для двух агентов на MT-графе.

Пусть дано частичное решение $PPS^{(i)}=\langle\pi^{(i)}, t^{(i)}\rangle$ и секция $e_j^{(i)}\in\pi^{(i)}$. Далее в целях упрощения верхние индексы не используются. $g$-значением

секции $e_j$ является сумма длин всех секций, составляющих частичный путь от вершины s до вершины $sp(e_j)$, а также задержка $t$: $g(e_j, PPS) = t + len(e_1) + len(e_2) + \ldots + len(e_{j-1})$. Пусть дана некоторая точка $p$, принадлежащая секции $e_j$. $g$-значением этой точки будем называть величину $g(p, e_j, PPS)=g(e_j, PPS)+dist(sp(e_j), p)$.

Две секции, принадлежащие различные частичным решениям являются потенциально конфликтными, если их пересечение (как отрезков) не пусто, то они. Секции являются конфликтными, если они являются потенциально конфликтными и существует точка, принадлежащая обеим секциям, такая что $g$-значения для обоих путей до этой точки равны. Два частичных решения являются конфликтными, если они содержат хотя бы одну пару конфликтных секций. Потенциальное решение является неконфликтным решением, если все его частичные решения являются неконфликтными

Задача согласования конфликтных состоит в преобразовании множества частичных решений, содержащих конфликты, в неконфликтное решение.

## 2 Типизация конфликтов и процедура их выявления

Конфликты могут быть разделены на три типа: пересечение, лобовое столкновение, преследование (см. Рис.2). Конфликты типа «пересечение» возникают между неколлинеарными секциями, которые пересекаются в одной точке. Если $g$-значения в этой точке для обоих путей равны – то эти секции конфликтны. Конфликты второго типа возникают, когда две секции коллинеарны и разнонаправленны. Конфликты третьего типа возникают между секциями, которые являются коллинеарными и сонаправленными.

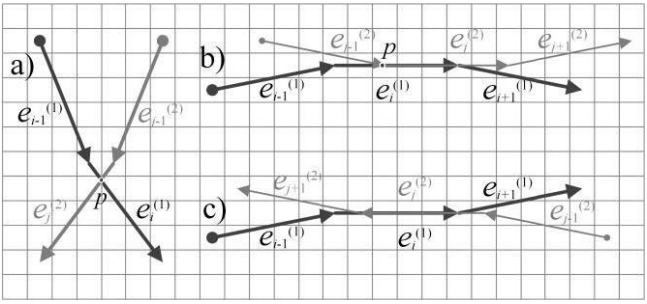



Рис.2 Типы конфликтов a) пересечение, b) преследование c) лобовое столкновение.

Пусть даны два частичных решения $PPS^{(1)}=\langle\pi^{(1)}, t^{(1)}\rangle$, $PPS^{(2)}=\langle\pi^{(2)}, t^{(2)}\rangle$, и секции $e_i^{(1)}\in\pi^{(1)}$, $e_j^{(2)}\in\pi^{(2)}$, которые необходимо проверить на конфликтность. Заметим, что секции не могут иметь конфликт, если

расстояние между концами секций больше чем сумма их длин. Если же это не так, то предлагается следующая процедура проверки на наличие конфликта.

Сначала необходимо проверить секции на коллинеарность. Если они не коллинарны, то возможен конфликт только первого типа. Используя геометрические формулы, можно найти точку пересечения, если она существует и посчитать g-значения для обоих путей. Строго говоря, чтобы считать две секции конфликтными, g-значения должны совпадать. Однако с практической точки зрения целесообразно использовать следующее выражение: $|g(p, e_i, PPS^{(1)}) - g(p, e_j, PPS^{(2)})| < r$, где r – минимальный безопасный радиус, который равен, к примеру, расстоянию между центрами двух горизонтально смежных вершин.

Если же две секции коллинеарны и находятся на одной прямой, необходимо проверить их на сонаправленность. Если секции разнонаправленны, между ними возможен конфликт второго типа. Необходимо проверить следующие соотношения: (1) $g(sp(e_i^{(1)}), PPS^{(1)}) + \text{dist}(sp(e_i^{(1)}), ep(e_j^{(2)})) \le g(ep(e_j^{(2)}), PPS^{(2)})$; (2) $g(sp(e_j^{(2)}), PPS^{(2)}) + \text{dist}(sp(e_j^{(2)}), ep(e_i^{(1)})) \le g(ep(e_i^{(1)}, PPS^{(1)})$. Если они выполняются, значит существует точка, принадлежащая обеим секциям, *g*-значение которой равно для обоих путей. В случае, когда секции сонаправленны, возможен конфликт третьего рода. Чтобы его проверить необходимо взять стартовую вершину секции, которая принадлежит общей части обеих секций, и посчитать её g-значения для обоих путей. Если они равны, значит эти две секции имеют конфликт.

Все вышеописанные проверки могут быть скомбинированы в одну функцию *FindFirstConflict*, которая станет одной из главных частей алгоритма разрешения конфликтов, который будет описан далее.

## 3 Локальное перепланирование

Пусть даны два частичных решения $PPS^{(1)}=\langle\pi^{(1)}, t^{(1)}\rangle$, $PPS^{(2)}=\langle\pi^{(2)}, t^{(2)}\rangle$, и секции $e_i^{(1)}\in\pi^{(1)}$, $e_j^{(2)}\in\pi^{(2)}$, которые являются конфликтными. Идея локального перепланирования заключается в том, чтобы построить незначительно отличающийся путь, изменив секции $e_j^{(2)}$ и $e_{j+1}^{(2)}$. Так как все пути являются Δ-путями, то можно предложить следующий подход к локальному перепланированию.

Сначала необходимо определить какие вершины находятся на расстоянии Δ от вершины $e_j$ (для этого можно воспользоваться алгоритмом [Pitteway, 1985]). Обозначим такие вершины как *CIRCLE*. Вершины $a_{kl}\in CIRCLE$, которые не удовлетворяют заранее заданному ограничению на максимальный угол отклонения $\boldsymbol{\alpha_m}$, удаляются из *CIRCLE*. Удаляя такие вершины, мы отсекаем резкие смены направления. После того, как набор

вершин-кандидатов найден, из него выбирается вершина $a_{kl}$ , такая что секции $e_{new1}=\langle ep(e_{j-1}), a_{kl}\rangle$ и $e_{new2}=\langle a_{kl}, ep(e_{j+1})\rangle$ проходимы. Это и есть секции обхода (см. Рис.3).

Обозначим данную процедуру как *ComputeLocalDetour*. Она будет использоваться в дальнейшем при построении алгоритма разрешения множества конфликтов.

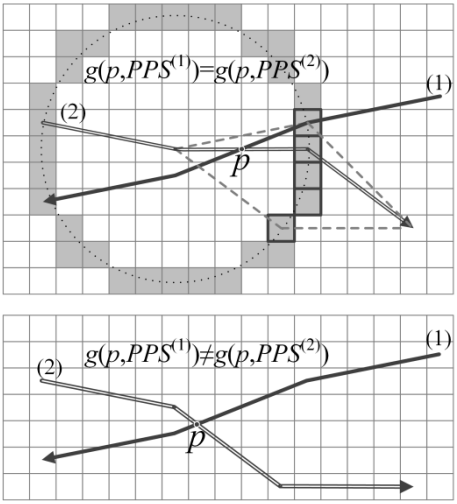



Рис.3 Процедура локального перепланирования. a) Вершины, формирующие Δ-окружность отмечены серым. Жирным отмечены вершины, удовлетворяющие ограничениям на максимальный угол отклонения. b) Результат процедуры перепланирования.

## 4 Алгоритм разрешения конфликтов

На вход алгоритму подается четверка <***PS***, **Δ**, $\boldsymbol{\alpha_m}$, ***wait***>, где ***PS*** – это начальное потенциальное решение (набор путей с нулевыми задержками), Δ – размер секции, $\boldsymbol{\alpha_m}$ **–** максимальный угол отклонения для функции *ComputeLocalDetour* и параметр ***wait*** **–** значение задержки (применяется в том случае, когда локальное перепланирование не смогло решить конфликт).

Псевдокод алгоритма представлен на Рис.4.

**Algorithm 1** Conflicts Resolution

**Input**: ***PS***, Δ, **α**, ***wait***; **Output**: ***PS***$' \in$ NoCON

1: $\{HEAD, TAIL\} \leftarrow FormHeadAndTail(\boldsymbol{PS})$
2: **while** $TAIL \neq \varnothing$
3: $PPS^{(cur)}$=argmin$_{PPS \in TAIL}$ $NumberOfConficts(PPS, TAIL \cup HEAD)$
4: $TAIL$.remove($PPS^{(cur)}$)
5: **while** ($\{e_v^{(cur)}, e_w^{(k)}\} \leftarrow FindFirstConflict(PPS^{(cur)}, HEAD)) \neq \varnothing$
6: $\pi_{new} \leftarrow ComputeLocalDetour(PPS^{(cur)}, e_v^{(cur)}, PPS^{(k)}, \Delta, \boldsymbol{\alpha})$
7: **if** $\pi_{new} = \pi^{(cur)}$
8: $t^{(cur)}$+=***wait***
9: **else**
10: $\{e_t^{(new)}, e_s^{(m)}\} \leftarrow FindFirstConflict(PPS^{(new)}, HEAD)$
11: **if** $\{e_t^{(new)}, e_s^{(m)}\} = \varnothing$ or $t>v$
12: $\pi^{(cur)} = \pi_{new}$
13: **else**
14: $t^{(cur)}$+=***wait***
15: $HEAD$.add($PPS^{(cur)}$)
16: **return** $HEAD$

Рис.4 Псевдокод алгоритма разрешения конфликтов.

Основная идея предлагаемого алгоритма заключается в разделении начального решения на два подмножества– *HEAD* и *TAIL*. В *HEAD* содержатся частичные решения, которые не конфликтуют между собой, а в *TAIL* – все остальные частичные решения, которые имеют, по крайней мере один конфликт с элементами из множества *HEAD*. На каждом шаге алгоритм выбирает частичное решения из *TAIL*, устраняет все конфликты, которые это решение имеет с решениями из *HEAD*, удаляет его из *TAIL* и добавляет в *HEAD*. Таким образом, на каждой итерации количество частичных решений в множестве *HEAD* увеличивается и в конечном итоге оно будет содержать все частичные решения. Т.к. все частичные решения множества *HEAD* не конфликтуют между собой, будет найдено искомое неконфликтное решение.

Первый шаг алгоритма заключается в формировании множеств *HEAD* и *TAIL* по исходным данным. После того, как эти множества сформированы, начинается основной цикл работы алгоритма (3-14). На каждой итерации выбирается потенциальное частичное решение с самым высоким приоритетом (приоритет определяется по количеству конфликтов, которое имеет это решение с множеством *HEAD*) из *TAIL* - $PPS^{(cur)}$ и удаляется из *TAIL*. В процессе работы цикла (6-13) устраняются все конфликты, и решение добавляется в *HEAD*.

Для того чтобы понять, что $PPS^{(cur)}$ не имеет конфликтов с *HEAD*

используется функция *FindFirstConflict*, которая возвращает пустое множество в случае, когда конфликтов между $PPS^{(cur)}$ и *HEAD* нет. Если же конфликты существуют, то функция вернет первый найденный конфликт.

После того как конфликт найден, алгоритм пытается его решить с помощью функции *ComputeLocalDetour*. Если изменить путь не удалось, задержка $t^{(cur)}$ увеличивается на значение ***wait***. Если функции *ComputeLocalDetour* удалось изменить путь, то необходимо проверить, не появился ли новый конфликт в текущей или более ранней секции. Если нет, считаем, что конфликт устранен и сохраняем новый путь. Если же возник конфликт в более ранней секции, то такое решение не используется по следующий причине. Допустим, был найден конфликт $con_a=\{e_v^{(cur)}\in\pi^{(cur)}, e_w^{(k)}\in\pi^{(k)}\}$. После того, как путь был локально перепланирован, был найден новый конфликт $con_b=\{e_t^{(new)}\in\pi^{(new)}, e_s^{(m)}\in\pi^{(m)}\}$ и при этом $t\leq v$. После этого, алгоритм пытается решить конфликт $con_b$ и снова создает конфликт $con_a$. В результате возникает цикл, который не улучшает решение. Поэтому, в случае, когда после локального перепланирования был найден конфликт в той же самой или более ранней секции, алгоритм отменяет изменения, сделанные функцией *ComputeLocalDetour*, и добавляет задержку. В отличие от локального перепланирования, задержка не приводит к возникновению циклов, так как задержка работает только в одну сторону, то есть, задерживая одного агента несколько раз подряд невозможно создать конфликты, которые были у этого же агента с меньшим числом задержек.

## 5 Экспериментальные исследования

Тестирование проводилось на персональном компьютере Intel Q8300 2.5GHz, 2Gb RAM. В качестве заданий использовались фрагменты карт городской местности (Москвы), полученные из открытой базы данных OpenStreetMaps (www.openstreetmaps.org). Было использовано 100 фрагментов размером 1347x1347 м, каждый из которых был преобразован в MT-граф 501x501 вершин (непроходимыми считаются вершины, соответствующие зданиям).

Для каждого фрагмента было сгенерировано 4 задания 2-х разных типов. В случае заданий первого типа, стартовые вершины расположены случайном образом на краях карты, а целевые вершины – на противоположных. Задания второго типа характеризуются тем, что все стартовые вершины расположены в области 50x50 на одном из краев карты, а все целевые вершины – в области 50x50 на противоположной стороне.

При тестировании использовались следующие значения параметров $\Delta=5$, $\boldsymbol{wait}=5$, $\boldsymbol{\alpha_m}=25$. В качестве алгоритмов планирования траектории

были использованы Theta* и LIAN.

В таблице 1 представлена подробная статистика, касающаяся процесса разрешения конфликтов.

Таблица 1. Статистика разрешенных конфликтов.

| | Theta* | | LIAN | |
|---|---|---|---|---|
| | Тип-1 | Тип-2 | Тип-1 | Тип-2 |
| Агенты | | | | |
| Задержаны | 12.07 | 64.685 | 15.3 | 72.335 |
| Перепланированы | 14.73 | 58.565 | 7.77 | 64.09 |
| Не изменены | 79.54 | 34.33 | 80.56 | 26.44 |
| Количество разрешенных конфликтов | | | | |
| Задержкой | 21.025 | 746.085 | 26.575 | 1176.19 |
| Перепланированием | 30.935 | 1442.185 | 12.7 | 638.3 |

В случае заданий первого типа в среднем 80% агентов вообще не подвергаются никаким изменениям, в то время как для разрешения всех конфликтов в заданиях второго типа требуется остановить и перепланировать 60-70% всех агентов. Так же видно, что количество конфликтов по секциям в несколько десятков раз выше у второго типа заданий.

В таблице 2 представлены результаты эксперимента, касающиеся времени работы и стоимости решения после обоих этапов планирования.

Таблица 2. Время работы алгоритма и стоимость найденного решения.

| | | Theta* | | LIAN | |
|---|---|---|---|---|---|
| | | Тип-1 | Тип-2 | Тип-1 | Тип-2 |
| Этап-1 | Время(с) | 7.3783 | 6.9426 | 9.5827 | 5.387 |
| | Стоимость | 46926 | 46722 | 49636 | 49651 |
| Этап-2 | Время(с) | 0.1466 | 0.804 | 0.1441 | 0.5926 |
| | Стоимость | 47034 (+0.23%) | 50477 (+8.04%) | 49772 (+0.27%) | 55566 (+11.91%) |

Полученные результаты свидетельствуют о том, что для заданий первого типа ухудшение стоимости решения составляет менее 1%, а для заданий второго типа – 10-15%. Такая большая разница связана с тем, что в заданиях второго типа возникает значительно большее число конфликтов, так как все агенты фактически движутся вместе из одной области в другую, в результате чего возникают конфликты на протяжении всего пути.

## Заключение

В работе рассмотрена задача согласования множества конфликтных траекторий возникающих при планировании перемещений группы интеллектуальных агентов на примере беспилотных летательных аппаратов, совершающих маловысотный полет в городских условиях. Предложены процедуры идентификации конфликтов и их разрешения путем локального перепланирования. На основе этих процедур разработан новый алгоритм, который позволяет разрешать конфликты и строить совокупность неконфликтных траекторий. Экспериментальные исследования показали его применимость к задаче, описанной в работе. В дальнейшем планируется теоретическое исследование разработанного алгоритма.

## Список литературы

**[Яковлев, 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.

**[Harabor, 2013]** Daniel Harabor and Alban Grastien. An Optimal Any-Angle Pathfinding Algorithm. In ICAPS, 2013.

**[Nash, 2007]** Alex Nash, Kenny Daniel, Sven Koenig, and Ariel Felner. Theta*: Any-Angle Path Planning on Grids. In Proceedings of the National Conference on Artificial Intelligence (Vol. 22, No. 2, p. 1177), 2007. Menlo Park, Calif.: AAAI Press.

**[Pitteway, 1985]** M. L. V. Pitteway. Algorithms of conic generation. In Fundamental algorithms for computer graphics, 219-237. Springer Berlin Heidelberg, 1985.

**[Silver, 2005]** David Silver. Cooperative pathfinding. In AIIDE, pages 117–122, 2005.

**[Wang, 2011]** Ko-Hsin Cindy Wang and Adi Botea. Mapp: a scalable multi-agent path planning algorithm with tractability and completeness guarantees. Journal of Artificial Intelligence Research, 42:55-90, 2011.

**[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.

**[Yu, 2013]** Yu, J. and LaValle, S.M., Structure and Intractability of Optimal Multi-Robot Path Planning on Graphs. In AAAI. 2013

**УДК 004.896**

# РЕШЕНИЕ ТРАЕКТОРНЫХ ЗАДАЧ ДЛЯ ГРУППЫ ЛЕТАТЕЛЬНЫХ АППАРАТОВ, ОСНОВАННОЕ НА ПРАВИЛАХ

М.В. Хачумов (*khmike@inbox.ru*)
ИСА ФИЦ ИУ РАН, Москва

**Аннотация**. Рассматриваются две траекторные задачи в условиях возмущений: преследование цели и следование по заданному маршруту. В качестве объектов выступают беспилотные летательные аппараты (ЛА), математические модели которых задаются передаточными функциями. Задача преследования цели заключается в сближении группы ЛА, имеющей некоторое случайное расположение, с целью и полета рядом с ней в течение заданного периода времени. Цель, имеющая меньшую скорость, стремится уклониться от преследователей. В задаче отработки заданного маршрута каждый аппарат следует по своей траектории полета, заданной движением соответствующей эталонной («имитационной») цели. В процессе решения задач ЛА применяют некоторое множество стратегий поведения в возмущенной среде, реализуемых правилами выбора углов ориентации и скоростей полета. Приведены примеры основных групп правил для преследователя и структурная схема моделирования процесса преследования. В экспериментальной части работы моделируются ситуации, характерные для решения указанных задач.[1]

**Ключевые слова**: беспилотный летательный аппарат, интеллектуальное управление, правила управления, групповое взаимодействие, преследование цели, моделирование.

## Введение

Одним из инструментов искусственного интеллекта, который активно применяется в системах управления, являются продукционные правила (продукции), реализующие принцип ситуационного управления. Достоинствами продукций являются: модульность – каждое правило не зависит от других; модифицируемость – правило может быть заменено или удалено вне зависимости от других; естественность –

[1] Работа выполнена при финансовой поддержке РФФИ (проект № 15-07-00925).

приближенность к тому к тому, как действует человек.

Укажем несколько характерных примеров управления роботизированными системами, основанного на правилах. В работе [Stroud, 1998] рассматривается отработка сложного движения самолета-перехватчика с помощью 15 основных правил. Автором [Прокопьев, 2010] предложена система управления ЛА, основанная на применении нечетких правил, реализуемых на нейронных сетях. Модель интеллектуального управления стыковкой космического аппарата, основанная на правилах, которые включают правила замыкания, переходов, выбора цели, управления, рассмотрена в работе [Смирнов, 2008]. Много исследований в этом направлении выполнено для управления движением наземных мобильных роботов, моделью которых является тележка [Ротштейн, 1999], [Евстигнеев, 2014].

В настоящей работе решаются задачи управления траекторным движением группы ЛА для преследования цели и следования по заданному маршруту в возмущенной среде. Решение задачи преследования динамической цели изучается в рамках теории дифференциальных игр [Friesz, 2010], [Lin, 2013], [Li, 2006]. Предложенные в этих работах математические постановки оптимизационных задач управления и существующие точные и приближенные методы их решения требуют больших затрат времени. В виду ограниченности возможностей вычислительных устройств и программного обеспечения автономных ЛА является актуальной задача построения простых в реализации, но эффективных алгоритмов управления, основанных на правилах. В работах [Дьяченко, 2012], [Li, 2012] отмечается эффективность группового применения ЛА при проведении мониторинга больших территорий и поиска целевых объектов за ограниченное время.

Ранее задачи преследования цели и следования по заданному маршруту решались автором для случая одного преследователя и одного убегающего [Хачумов, 2015], [Хачумов и др., 2015]. Полученные результаты позволили перейти вплотную к разработке стратегий поведения и экспериментальному моделированию процессов решения задачи для группы ЛА.

## 1 Постановка задачи

Рассмотрим две базовые задачи, имеющие прикладное значение.

*Задача 1 (Преследование цели).* Рассмотрим задачу преследования объекта $c$, обозначаемого дальше «цель», группой ЛА $P = \{p_1, ..., p_N\}$, $N \geq 2$. Верхними индексами $p$ и $c$ будем помечать переменные ЛА и

цели, соответственно. Объекты $p_i$ и $c$ осуществляют простое движение между соседними точками смены направления с постоянными скоростями $v_i^{(p)}$, $v^{(c)}$; углами тангажа $\theta_i^{(p)}(t)$, $\theta^{(c)}(t)$ и рыскания $\psi_i^{(p)}(t)$, $\psi^{(c)}(t)$, задающими ориентацию. Преследователю $p_i$ в момент времени $t_i$ известны: скорость $v^{(c)}(t)$; углы ориентации $\theta^{(c)}(t)$, $\psi^{(c)}(t)$; координаты цели $x^{(c)}(t), y^{(c)}(t), z^{(c)}(t)$. Аналогично, цель имеет полную информацию о преследователях. Пусть $\forall i, v_i^{(p)} \geq v^{(c)}$, $X_i(t)$ и $Y(t)$ − координаты ЛА $p_i$ и цели $c$, а $d(X_i(t), Y(t))$− расстояние между ними в момент времени $t$. Необходимо осуществить сближение группы ЛА, имеющей некоторое случайное расположение, с целью и полет рядом с ней в течение заданного времени наблюдения $T$. В дальнейшем ограничимся наличием у преследователей двух скоростей $v_i^{(p)} = (v^{(p1)}, v^{(p2)})$, причем $v^{(p1)} = v^{(c)}$, $v^{(p2)} > v^{(c)}$. В результате ветровой нагрузки возможно отклонение ЛА от своего маршрута, причем существенное.

Задача заключается в построении такого управления $(v_i^{(p)}(t), \theta_i^{(p)}(t), \psi_i^{(p)}(t))$ для каждого $p_i$ на временном отрезке $[0, T]$, что $\int_{t=0}^{T} d(X_i(t), Y(t))dt \to \min$ при ограничении: $\forall i, d(X_i(t), Y) \geq \varepsilon$, где $\varepsilon$ – допустимое значение сближения летательных аппаратов с целью.

*Задача 2 (Следование по заданному маршруту).* Пусть траектория летательного аппарата $p_i$ задана движением эталонной цели $c_i \in \{c_1, ..., c_N\}$ и представлена последовательностью опорных точек $(x_{ij}^{(c)}, y_{ij}^{(c)}, z_{ij}^{(c)})$, $j = 1, ..., M$. Известно желаемое время прохождения опорных точек $t_{ij}^{(c)}$, $i = 1, ..., N$, $j = 1, ..., M$ и всего маршрута в целом $T_i^{(c)}$. При этом ветровая нагрузка существенно влияет на движение ЛА, вызывая ее отклонение от заданного маршрута. Пусть $X_i(t)$ и $Y_i(t)$ − координаты $p_i$ и соответствующей эталонной $c_i$, а $d(X_i(t), Y_i(t))$ − расстояние между ними в момент времени $t$.

Задача заключается в построении такого управления $(v_i^{(p)}(t), \theta_i^{(p)}(t), \psi_i^{(p)}(t))$ для каждого ЛА в группе на временном отрезке

$[0, T_i^{(p)}]$, что $\int_{t=0}^{T_i^{(p)}} d(X_i(t), Y_i(t))dt \to \min$.

Предлагается решение поставленных задач, основанное на действиях, имитирующих поведение пилота в условиях ветровых возмущений. В работе [Khachumov, 2015] показано что решение задачи определения точки встречи ЛА и цели может быть получено геометрическим способом на основе построения сферы Аполлония. В случае, когда цель меняет направление движения, применяется стратегия «параллельного сближения», основанная на перепланировании движения преследователя путем вычисления новых углов атаки. В настоящей работе решается задача построения системы продукционных правил и моделирование группового преследования цели.

## 2 Описание системы управления ЛА на основе продукционных правил

В работе [Хачумов, 2015] представлена схема интеллектуальной системы управления автономным ЛА, в которой учтена модель ветровых возмущений и отмечена существенная роль выбора правил управления.

Приведем набор правил для преследователя $p_i$. Запись вида $(A, B, C...)$ обозначает «известно значение $A$» и «известно значение $B$» и «известно значение $C$»…. Спискам добавляемых и удаляемых фактов предшествует служебное слово «ТО», а выражение вида $x := a$ означает «в текущем состоянии базы данных присвоить переменной $x$ значение $a$». Индекс $t$ при переменной означает привязку к текущему времени.

$F_1(d_i, x_i^{(p)}, y_i^{(p)}, z_i^{(p)}, x^{(c)}, y^{(c)}, z^{(c)}, \theta^{(c)}, \psi^{(c)}, v_i^{(p)}, v^{(c)})$ – функция, вычисляющая углы сближения $\theta_i'^{(p)}(t)$, $\psi_i'^{(p)}(t)$ для встречи с целью на сфере Аполлония; $F_2(\theta^{(c)}, \psi^{(c)}, v_i^{(p)}, \bar{v}_i^{(w)})$ – функция, вычисляющая угол следования за целью $\theta_i''^{(p)}(t)$, $\psi_i''^{(p)}(t)$ с учетом ветровой нагрузки [Хачумов и др., 2015]. Здесь $d_i$ – расстояние между $p_i$ и целью, $\bar{v}_i^{(w)}$ – вектор скорости ветра. Положим $\Delta t$ – продолжительность одного такта (шага) моделирования; $\varepsilon$ – минимальное расстояние, означающее, что произошло сближение с целью.

*Правила замыкания*. Правила осуществляют вычисление и подготовку необходимых для последующих расчетов данных, доопределяющих текущее состояние объекта управления.

ЕСЛИ $(x_i^{(p)}(t), y_i^{(p)}(t), z_i^{(p)}(t), x^{(c)}(t), y^{(c)}(t), z^{(c)}(t))$

ТО $d_i(t) := \sqrt{(x_i^{(p)}(t) - x^{(c)}(t))^2 + (y_i^{(p)}(t) - y^{(c)}(t))^2 + (z_i^{(p)}(t) - z^{(c)}(t))^2}$;

ЕСЛИ $(d_i(t), x_i^{(p)}(t), y_i^{(p)}(t), z_i^{(p)}(t), x^{(c)}(t), y^{(c)}(t), z^{(c)}(t), \theta^{(c)}(t), \psi^{(c)}(t),$
$v_i^{(p)}(t), v^{(c)}(t))$

ТО $\theta_i'^{(p)}(t) := F_1[\theta]$, $\psi_i'^{(p)}(t) := F_1[\psi]$;

ЕСЛИ $(\theta^{(c)}(t), \psi^{(c)}(t), v_i^{(p)}(t), \bar{v}_i^{(w)}(t))$

ТО $\theta_i''^{(p)}(t) := F_2[\theta]$, $\psi_i''^{(p)}(t) := F_2[\psi]$.

*Правила переходов.* Правила определяют переход системы в новое состояние в результате импульса управления с шагом $\Delta t$.

ЕСЛИ $(x_i^{(p)}(t), y_i^{(p)}(t), z_i^{(p)}(t), \bar{v}_i^{(w)}(t), \theta_i^{(p)}(t), \psi_i^{(p)}(t))$

ТО $x_i^{(p)}(t + \Delta t) := x_i^{(p)}(t) + v_i^{(p)}(t)\cos(\theta_i^{(p)}(t))\cos(\psi_i^{(p)}(t)) + \bar{v}_i^{(w)}(t)[x]$

$y_i^{(p)}(t + \Delta t) := y_i^{(p)}(t) + v_i^{(p)}(t)\cos(\theta_i^{(p)}(t))\sin(\psi_i^{(p)}(t)) + \bar{v}_i^{(w)}(t)[y]$;

$z_i^{(p)}(t + \Delta t) := z_i^{(p)}(t) + v_i^{(p)}(t)\sin(\theta_i^{(p)}(t)) + \bar{v}_i^{(w)}(t)[z]$;

Правила выбора цели

ЕСЛИ $(d_i(t) > \varepsilon)$ ТО задача «сближение» := «активна»,

«следование» := «неактивна»;

ЕСЛИ $(d_i(t) \le \varepsilon)$ ТО задача «сближение» := «неактивна»,

«следование» := «активна».

Правила управления

ЕСЛИ (задача «сближение» = «активна»)

ТО $v_i^{(p)}(t) := v^{(p2)}(t)$, $\theta_i^{(p)}(t) := \theta_i'^{(p)}(t)$;

ЕСЛИ (задача «следование» = «активна»)

ТО $v_i^{(p)}(t) := v^{(p1)}(t)$, $\theta_i^{(p)}(t) := \theta_i''^{(p)}(t)$.

## 3 Система моделирования полета ЛА с учетом ветровой нагрузки

Общая схема моделирования процесса преследования цели ЛА в системе Simulink с учетом ветровых нагрузок приведена на Рис.1.

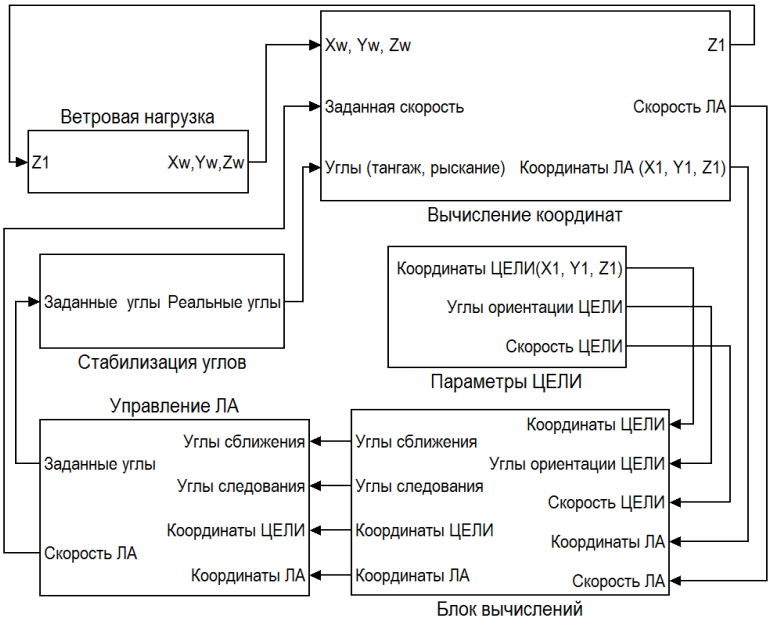



Рис. 1. Структурная схема моделирования преследования цели ЛА

Рассмотрим основные блоки структурной схемы.

*Блок «стабилизация углов».* На вход блока подаются углы ориентации, которые должен отработать ЛА с учетом возмущений. На выходе блок выдает текущий угол тангажа $\theta_i^{(p)}(t)$ и рыскания $\psi_i^{(p)}(t)$ ЛА $p_i$. Структурная схема стабилизации углов рассмотрена в работе [Khachumov, 2015].

*Блок «вычисления координат».* В принятой модели положение ЛА в текущий момент времени $t$ вычисляются на основе координат в предыдущий момент времени $t-1$ и известных параметров: заданной скорости $v_i^{(p)}(t)$, углов $\theta_i^{(p)}(t)$, $\psi_i^{(p)}(t)$, а также скорости ветра $\overline{v}_i^{(w)}(t)$. На выходе блок выдает реальную скорость и текущие значения координат $(x_i^{(p)}(t_i), y_i^{(p)}(t_i), z_i^{(p)}(t_i))$ ЛА.

*Блок «ветровая нагрузка».* Продольные и нормальные ветровые воздействия моделируются с помощью встроенных функций Simulink [Khachumov, 2015].

*Блок «параметры ЦЕЛИ»* выдает координаты $(x^{(c)}(t_i), y^{(c)}(t_i), z^{(c)}(t_i))$, углы ориентации $\theta^{(c)}(t)$, $\psi^{(c)}(t)$ и скорость $v^{(c)}(t)$ цели.

*Блок «вычислений»* выдает углы сближения $\theta_i'^{(p)}(t)$, $\psi_i'^{(p)}(t)$ и углы следования $\theta_i''^{(p)}(t)$, $\psi_i''^{(p)}(t)$.

*Блок «управления ЛА»* вырабатывает конкретные управления $\nu_i^{(p)}(t)$, $\theta_i^{(p)}(t)$, $\psi_i^{(p)}(t)$ в соответствии с текущим состоянием системы.

## 4 Экспериментальные исследования

Экспериментальные исследования описанных задач с использованием модели ЛА [Khachumov, 2015] в условиях возмущенной воздушной среды выполнены в системе MATLAB Simulink.

На Рис. 2 отмечены начальные (случайные) расположения ЛА и представлены траектории их движения для задачи преследования цели (*Задача 1*). Пройденное расстояние показано в километрах. Видно, что ЛА (светлые кривые) сначала сближаются с целью (темная кривая), а затем следуют за ней.

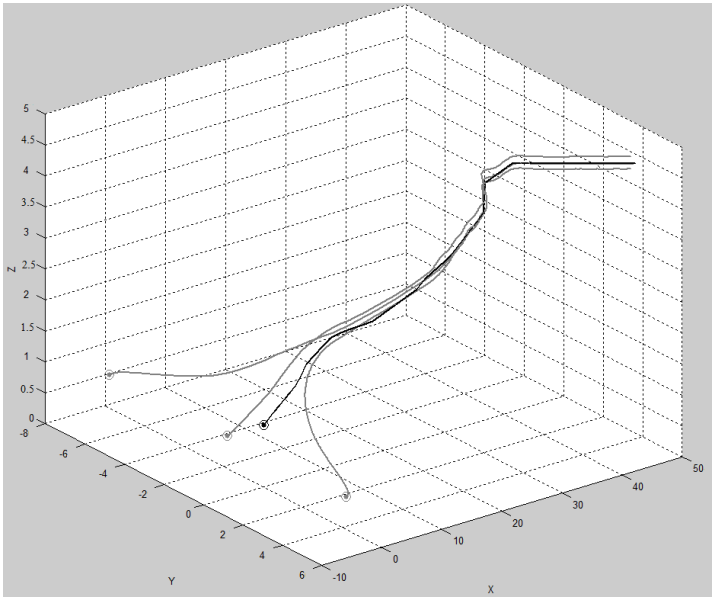



Рис. 2. Траектории движения цели и ЛА для примера *Задачи 1.*

На Рис. 3 представлены траектории движения ЛА для случая следования по заданному маршруту (*Задача 2*), при этом каждый ЛА (светлые кривые) преследует свою эталонную цель (темные кривые).

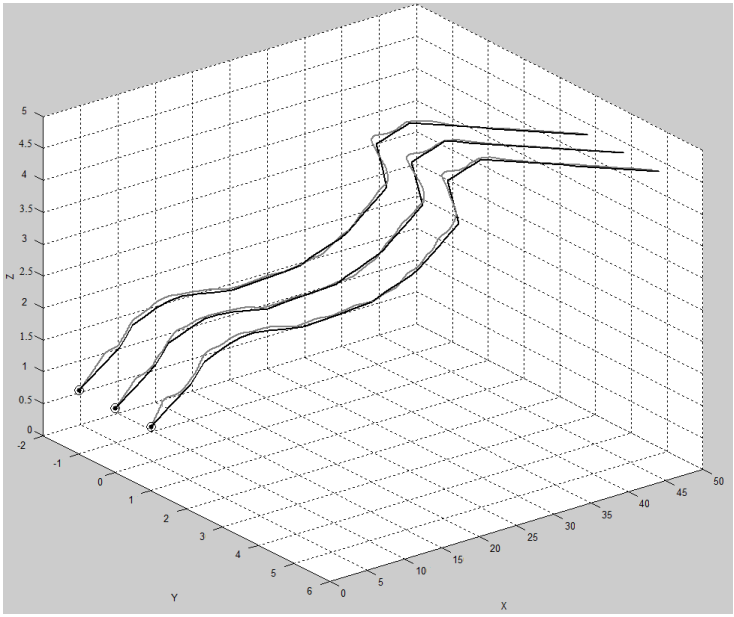



Рис. 3. Траектории движения цели и ЛА для примера *Задачи 2.*

Рис. 2-3 демонстрируют влияние ветровой нагрузки на реальную траекторию движения ЛА, однако заложенные в систему управления правила успешно справляются с поставленными задачами. Основным критерием качества управления служат интегральная оценка отклонения ЛА от заданного маршрута (от заданной цели) в процессе движения и визуальная оценка траектории полета участников сцены.

## 5 Заключение

В статье рассмотрены две траекторные задачи для группы ЛА, которые решались в условиях возмущенной воздушной среды. Выполненные эксперименты по моделированию процессов сближения ЛА с целью в системе Simulink, показали адекватность разработанных стратегий и реализующих их правил, которые задают естественное поведение объектов. Разработанный подход может быть использован для получения приемлемого по точности решения рассмотренных задач в условиях необходимости оперативных действий, вызванных отклонениями при ветровых воздействиях. Алгоритмы управления, основанные на правилах, целесообразно использовать на малых ЛА, имеющих ограниченные возможности бортовой вычислительной аппаратуры.

## Список литературы

**[Stroud, 1998]** D. Stroud. Adaptive Simulated Pilot Phillip // Journal of Guidance, Control, and Dynamics. 1998. Vol.21. No.2. pp.352-354.

**[Прокопьев, 2010]** Прокопьев И.В. Автоматизация системы автономного управления беспилотным летательным аппаратом // Труды Международного симпозиума «Надежность и качество», 2010, Т.1., с.420-422.

**[Смирнов, 2008]** Смирнов С.В. Модель интеллектуального управления космическими аппаратами средствами наземных станций командно-измерительных систем // Авиакосмическое приборостроение, 2008, № 6, с.42-49.

**[Ротштейн, 1999]** Ротштейн А.П. Интеллектуальные технологии идентификации. – Винница: УНИВЕРСУМ-Винница, 1999, 295 с.

**[Евстигнеев, 2014]** Евстигнеев Д.В. Системы управления интеллектуальных мобильных роботов в среде Dyn-Soft RobSim 5. Учебное пособие, 2014, 190 с. – http://robsim.dynsoft.ru/design3.pdf.

**[Дьяченко, 2012]** Дьяченко А.А. Задача формирования строя в группе БПЛА // Известия ЮФУ. Технические науки. – Таганрог: Южный федеральный университет, 2012, №3, с.22-30.

**[Li, 2012]** Li X. Swarm-inspired solution strategy for the search problem of unmanned aerial vehicles // PhD thesis, University of Warwick. 2010

**[Friesz, 2010]** Friesz T.L. Dynamic optimization and differential games // International Series in Operations Research & Management Science, Springer Science + Business Media, LLC, 2010, 502 p.

**[Lin, 2013]** Wei Lin. Differential games for multi-agent systems under distributed Information // A dissertation submitted for the degree of Doctor of Philosophy, University of Central Florida, 2013, 117 p.

**[Li, 2006]** Dongxu Li. Multi-player pursuit-evasion differential games // A dissertation submitted for the degree of Doctor of Philosophy, The Ohio State University, 2006, 151 p.

**[Khachumov, 2015]** Khachumov M.V., Abramov N.S., Makarov D.A.. Controlling Flight Vehicle Spatial Motion Along a Given Route // Automation and Remote Control. 2015. Vol.76. No.6. pp.1070-1080.

**[Хачумов, 2015]** Хачумов М.В. Решение задачи следования за целью автономным летательным аппаратом // Искусственный интеллект и принятие решений, 2015, №2, с.45-52.

**[Хачумов и др., 2015]** Абрамов Н.С., Хачумов М.В. Моделирование проводки по маршруту беспилотного летательного аппарата как задачи преследования цели // Авиакосмическое приборостроение, 2013, №9, 2013, с.9-22.

**УДК 004.021**

# АЛГОРИТМ КЛАСТЕРИЗАЦИИ КОЛЛЕКТИВА РОБОТОВ

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

**Аннотация**. В статье рассматривается вопрос применения механизмов роения для задач коллективной робототехники. Предлагается алгоритм, который инициализируется лидером роя и заключается в последовательном разделении роя на заданное количество кластеров и выборе лидера в каждом из полученных кластеров, используя только локальное взаимодействие. Процесс повторяется до тех пор, пока их число не станет равным заданному, при этом количество кластеров, получаемых за одну итерацию деления, зависит от количества соседей лидера[1].

**Ключевые слова**: статический рой, групповая робототехника, модели социального поведения, кластеризация

## Введение

В настоящий момент широкий интерес в области робототехники представляют вопросы, касающиеся коллективной робототехники (swarm robotics). Возможное практическое применение систем, состоящих из множества роботов, огромно: разведка [Tan et al., 2013], патрулирование [Bina, 2013], [Portugal et al., 2013], и т.д. Особый интерес представляет появление подобных систем, состоящих из роботов, которые имеют относительно простую конструкцию [Rubenstein et al., 2009], что должно привести к простоте масштабирования системы, высокой отказоустойчивости и т.д. и появлению в такой системе эмерджентных свойств [Yogeswaran et al., 2010].

С другой стороны, система из простых роботов обладает рядом ограничений, одно из которых – локальный характер взаимодействия друг с другом. В свою очередь это приводит к некоторым особенностям реализации механизмов управления коллективом и передачи информации между его членами.

В основе реализации механизмов коллективного управления могут

[1] Работа выполнена при финансовой поддержке РНФ (проект № 16-11-00018) и РФФИ (проект № 15-01-07900 и проект №16-29-04412).

лежать как автоматные модели [Цетлин, 1969], [Варшавский и др., 1984], так и многоагентные системы, эволюционные модели и т.д. В этом ряду можно выделить концепцию, которая подразумевает создание систем групповой робототехники, используя модели социального поведения, которая достаточно подробно описана в [Карпов, 2016].

Одним из интересных механизмов социального поведения, который наблюдается у многих видов муравьев и пчел и может быть полезен в коллективной робототехнике, является *роение*. Роение – размножение пчелиной семьи путем отделения от материнской семьи половины пчел с маткой и трутнями [Кокорев и др., 2005]. Если рассматривать пчел, то причины, способствующие старту роения, являются большое число молодых особей, не занятых работой в улье, старая матка, плохая вентиляция гнезда и его перегрев, переполнение его печатным расплодом и т.д [Тамбовцев и др., 2004].

Примером применения роения в робототехнике может являться ситуация, когда коллективу необходимо выполнить ряд параллельных действий. Действительно, наблюдая за пчелами или муравьями, можно легко увидеть, что у них есть функциональная дифференциация. Часть занимается разведкой, часть обустройством улья/муравейника и т.д. Ситуации, когда рой целиком занят одной задачей, если и встречаются, то крайне редко. Таким образом, существует необходимость в механизме разделения роя в качестве дополнения к алгоритму функциональной дифференциации.

Кроме того, рой роботов может работать в условиях, когда каналы локальной связи работают нестабильно или данную связь сознательно подавляет противник. В таком случае информация от периферийных узлов может доходить до лидера с потерями или теряться вовсе. При этом, чем дальше узел-отправитель от лидера, тем больше узлов ее ретранслирует, тем вероятнее ее потери или повреждения. Следовательно, имеет смысл не собираться в таких условиях в большие группы, а если рой уже обладает большим числом элементов – разделиться.

Вообще говоря, вопрос интеграции механизма роения в систему управления коллективом роботов не является новым. В данной области эта проблема называется *кластеризацией*.

В работах [Chen et al., 2012], [Gross et al., 2009] представлен механизм, основанный на гранулярной конвекции или на *эффекте бразильского ореха*. Основная идея работы – группировать роботов по принципу случайного движения вокруг общей точки притяжения и отталкивания между соседями. Очевидно, что во время выполнения данного алгоритма тратится дополнительная энергия на передвижение роботов, что может быть критично в ряде задач.

В работе [Caro et al., 2012] представлена интересная реализация

алгоритма кластеризации, основанная на механизме передачи жетонов (token'ов) между роботами. Наличие жетона определяет принадлежность к кластеру. Алгоритм содержит в себе элементы, используемые в бисекции и балансировки нагрузки в сетях.

Идеологическим продолжением данной работы является [Cruz et al., 2016], где предложена модификация предыдущего алгоритма, позволяющая разделять на произвольное количество кластеров.

Кроме того, в работе [Каляев и др, 2009] достаточно подробно описывается процесс кластеризации. Выделяется иерархическая кластеризация, причем иерархия может выстраиваться двумя путями: "сверху-вниз" и "снизу-вверх". Если механизм иерархической кластеризации идет по пути объединения отдельных роботов в кластеры, то она называется "снизу-вверх ". В случае если производится дробление большой группировки роботов на более мелкие, то это кластеризация "сверху-вниз". Также описаны механизмы создания непересекающихся кластеров постоянного состава, как для гетерогенных, так и для гомогенных групп роботов. Рассматриваются проблемы перераспределения роботов между кластерами и динамической кластеризации – процесс инициализации кластеров и их роста.

В основном в работах рассматривается проблема кластеризации в децентрализованных системах [Caro et al., 2012], [Cruz et al., 2016]. Однако в ряде задач целесообразно использовать централизованные системы, например задачи, где необходима высокая степень координации и контроля. При этом централизованная система также может решать задачи, например, функциональной дифференциации, где так или иначе необходимо проводить кластеризацию.

В связи с этим имеет смысл использовать особенности таких систем для реализации механизма деления. К примеру, наличие лидера может существенно упростить данную задачу, так как именно он может иметь полномочия формировать кластеры под конкретную задачу.

Однако, как уже было сказано, есть существенное ограничение в виде локального характера взаимодействия между роботами, т.е. он может общаться только со своими соседями. Следовательно, даже лидер не знает структуры и топологии роя, которым он командует. Это приводит к некоторым особенностям реализации механизма кластеризации в такой системе.

В работе предлагается достаточно простой алгоритм кластеризации коллектива роботов, который обладает лидеров, где каждый робот использует локальное взаимодействие. Лидер формирует зачатки кластера, но так как структура и топология роя ему неизвестны, зачатками становятся его соседи, которые стараются присоединить к себе своих соседей и т.д. Таким образом, легко формируются кластеры, число

которых меньше, либо равно числу соседей лидера. В случае если кластеров должно быть больше, вместе с формированием их зачатков, по числу соседей лидера им передается информация о необходимом делении после формирования кластера. Затем процедура повторится уже для сформированной группы роботов. Более всего данная процедура напоминает иерархическую кластеризацию “сверху-вниз”, описанную в [Каляев и др, 2009]. В работе показана техническая реализация данной процедуры в условиях наличия лидера и локального взаимодействия.

Отдельно стоит отметить, что, не смотря на то, что понятия “коллектив роботов” и “рой роботов” различны [Карпов, 2016], это нисколько не должно влиять на механизм роения, т.к. он должен работать как в коллективе, так и в рое роботов.

Кроме того, в работе используются термины “робот” и “узел”, которые являются синонимами в контексте данной задачи: подразумевается, что, в конечном итоге, алгоритм должен работать в коллективе роботов, а термин “узел” чаще используется при описании работы алгоритма в качестве некоторой абстракции. Аналогично обстоит дело с терминами “фрагмент”, “часть”, “кластер” и “кластеризация”, ”роение”.

## 1 Постановка задачи

Рассмотрим задачу: существует N роботов, взаимодействие которых друг с другом носит локальный характер, т.е. каждый робот “знает” о существовании только себя и своих ближайших соседей. Общая структура роя неизвестна никому, однако, средствами локального взаимодействия имеется возможность собрать косвенные данные, например, число элементов в нем. В рое выбирается лидер, используя, например, алгоритм, описанный в [Карпов и др., 2015].

Кроме того, предполагается, что в процессе кластеризации топология роя не меняется, т.е. его можно представить в виде структуры под названием статический рой - полученная в некий момент времени схема соединения роботов по каналам связи [Карпов, 2013]. При этом совсем не обязательно допускать, чтобы роботы были неподвижны – важно чтобы сохранялась схема соединений друг с другом.

Таким образом, задача звучит следующим образом: необходимо провести процедуру роения в статическом рое, разделив ее на M заданных частей.

## 2 Алгоритм кластеризации

Решение этой задачи осуществляется поэтапно: лидер формирует зачатки кластера и, при необходимости, передает информацию о дальнейшей кластеризации. Затем в каждом кластере выбирается свой

лидер, который уже имеет информацию о том, надо ли делиться дальше. Если такая необходимость есть, то процесс повторяется.

В первую очередь необходимо чтобы в рое был лидер, который инициирует процесс кластеризации. Механизм выбора лидера, который использовался в этой работе базируется на локальной передаче сообщений от периферийных узлов в центр роя. Это позволяет подсчитать количество уникальных соседей узла, в том числе и тех, с которыми он связан через другие узлы. Такой механизм позволяет выбирать лидера, который располагается достаточно близко к топологическому центру роя, что обеспечивает приемлемое время, которое необходимо сообщению от любого периферийного узла, чтобы дойти до лидера.

Алгоритм кластеризации представляет собой несколько процедур деления/выбора лидера, количество которых зависит от количества частей, на которые необходимо поделить рой и количества соседей у лидера, которым он может отправить сообщение о начале роения. Алгоритм начинается с сообщения, которое отправляет лидер своим соседям. При этом, если количество соседей больше или равно количеству частей, на которые рой необходимо разделить, то лидер произвольно выбирает кому он отправит данное сообщение. Каждому из выбранных соседей сообщается уникальный номер части, на которые делится рой. В свою очередь данные узлы отправляют данное сообщение с уникальным номером своим соседям и т.д. Если узел уже принадлежит какой-либо части, то он не может принимать сообщения от частей с другим уникальным номером, что позволяет закончить работу алгоритма разделением роя на заданное количество частей.

В случае если разделить рой необходимо на части, количество которых больше, чем количество соседей лидера, то алгоритм работает так, как если бы они были равны. Однако недостающие разделения запоминаются соседями лидера, которые передадут их своему новому лидеру. В дальнейшем, когда рой разделится первый раз и в каждой из частей появится лидер, инициирует новое разделение и т.д.

Например, есть рой, состоящий из 10 особей, у лидера 3 соседа и надо разделить его на 8 частей. Сначала рой разделится на 3 части, в каждой из которых будет храниться информация о том, что необходимо делиться дальше. Первоначально, она будет храниться только у соседей лидера, но в процессе разделения ее будут передавать всем узлам формирующейся части роя. В данном случае два соседних узла будут знать, что из их фрагментов должно получиться по три части в каждой, а третьему соседу известно, что его фрагмент должен разделиться надвое. Когда в каждом фрагменте появится свой лидер, то он инициирует новое разделение, в соответствии с данной информацией. Псевдокод алгоритма представлен ниже.

Алгоритм. 1. Алгоритм кластеризации

**Алгоритм К1(М). Кластеризация**
A – элемент роя (узел).
N – количество соседей узла A.
$C_i$ – i-й сосед узла.
M – количество частей, на которые необходимо разделить рой.
$P_c$ – номер части роя, к которой принадлежит узел-сосед.
$P_A$ - номер части роя, к которой принадлежит узел A.
l – целочисленный результат деления M на N.
k – остаток от деления M на N.
$Mem_c$ – информация о необходимости дальнейшего деления, хранящаяся в узле C.
$Mem_A$ – информация о необходимости дальнейшего деления, хранящаяся в узле A.
**Процедура К1.**
**if** M>N **then** – Если число частей, на которые необходимо разделить больше числа соседей лидера.
  l = M/N
  k = M%N
**end if**
**if** A- лидер **then**
  **for** i:=1 **to** N [**step** 1] **do** – Цикл по соседям лидера
    **if** $C_i$ еще не принял сообщение о роении **then**
      $C_i$ принимает запрос о роении.
      $P_c$ = i
      **if** M>N **and** i <> N **then** – Если число частей, на которые необходимо разделить больше числа соседей и есть еще соседи, которым сообщение не передано.
        $Mem_c$= l
      **end if**
      **if** M>N **and** i == N **then** - Если число частей, на которые необходимо разделить больше числа соседей и это последний сосед, которому еще не передали сообщение
        $Mem_c$= k
      **end if**
    **end if**
  **end for**
  A – не лидер
**end if**
**if** A- принял сообщение **then**
  **for** i:=1 **to** N [**step** 1] **do**

**if** $C_i$ еще не принял сообщение о роении **then**
$C_i$ принимает запрос о роении.
$P_c = P_A$
$Mem_c = Mem_A$
**end if**
**end for**
Выбор лидера
**end if**

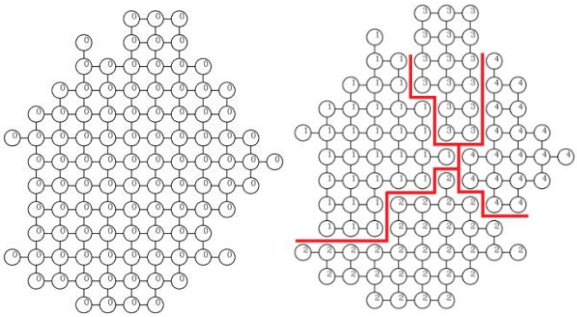
## 3 Вычислительные эксперименты

Алгоритм был реализован в виде программы на языке C++. Входными данными алгоритма является количество роботов, составляющих рой, который необходимо разделить и число кластеров, на которые надо его разделить. Их число варьировалось от 50 до 200 с шагом в 25. Таким образом, было проведено 7 симуляций, по 100 итераций в каждом, где в каждой итерации случайно создавалась своя конфигурация роя с заданным количеством роботов.

Затем осуществлялись выборы лидера, и он выбирал соседей, которые будут формировать кластер. Рой делится на четыре части (рис. 1). Во всех случаях кластеризация прошла успешно. Эксперимент показал, что время роения линейно зависит от максимального расстояния между лидером и любым периферийным узлом(1):

$$T=O(\max(S(L,P_1), S(L,P_2),\ldots,S(L,P_n))), \quad (1)$$

где T – время роения, S(x,y) – расстояние, между x и y, L-лидер, $P_n$-периферийный узел.



а) б)

Рис. 1. а) – Рой до разделения, б) - Рой после разделения. Границы фрагментов отмечены линиями

Кроме того, установлено, что основной вклад во время работы алгоритма роения вносит механизм повторного выбора лидера в новом рое (рис. 2).

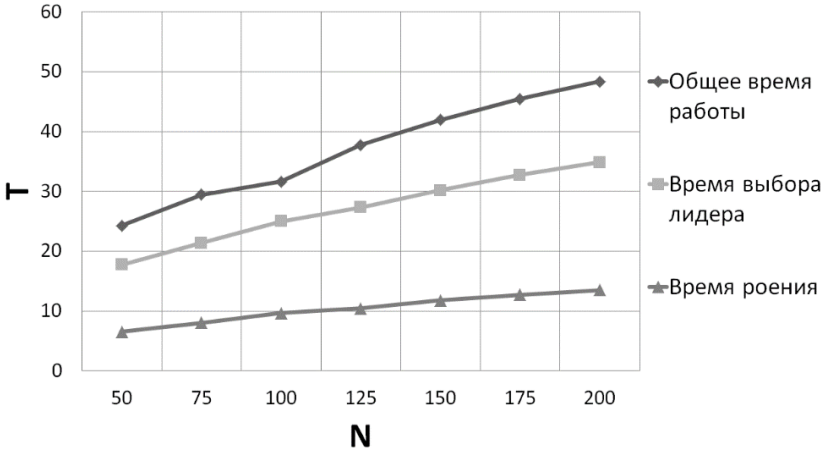



Рис. 2 Зависимость среднего времени выполнения (T) от количества роботов (N).

При разном начальном количестве роботов их число после роения в каждом кластере примерно одинаково для каждого кластера (рис. 3).

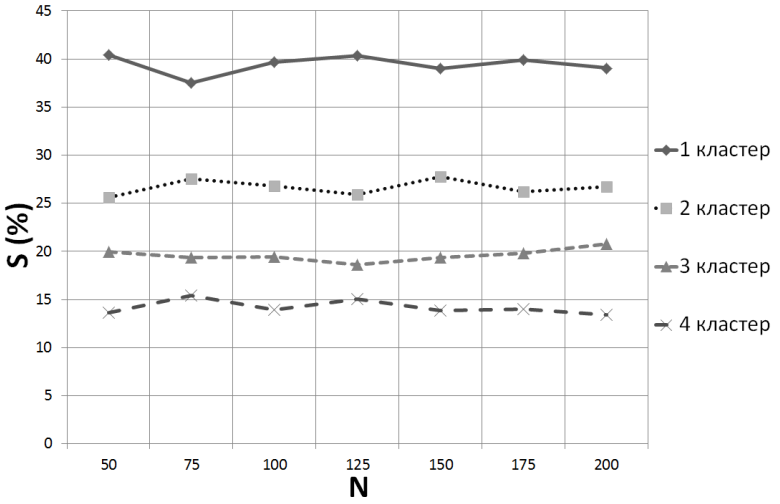



Рис. 3 Процентное соотношение числа роботов в каждом кластере S в зависимости от числа роботов N. 100% - число роботов в начале роения.

При разделении на небольшое количество фрагментов, время работы алгоритма зависит только от числа роботов в рое. Поэтому, для наиболее эффективной работы алгоритма необходимо, чтобы выполнялось неравенство (2):

$$N>=M, \quad (2)$$

где N – количество соседей лидера, M – количество фрагментов

## Заключение

Таким образом, был предложен алгоритм кластеризации, который может быть полезен для задач коллективной робототехники, в частности, для задач функциональной дифференциации. При этом стоит отметить, что данный алгоритм идеологически наиболее подходит для разделения роя на небольшое число фрагментов.

Это обуславливается тем, что алгоритму требуется повторно выбирать лидера в каждом фрагменте, что приводит к увеличению времени работы алгоритма при большом количестве фрагментов.

С другой стороны, с ростом числа кластеров накладные расходы на выбор лидера в каждом новом кластере должны сократиться, так как число роботов внутри них будет падать.

Кроме того, стоит отметить, что используемый алгоритм выбора лидера, также написанный автором, в процессе моделирования оказывал существенно влияние на результаты работы процедуры кластеризации. Данное влияния заключается не столько во времени, которое затрачивается на выбор лидера, сколько в том, какой именно узел будет выбран лидером. Если в самом начале лидер находился достаточно близко к центру роя, то после появления кластеров их лидеры не всегда обладали таким свойством, что часто приводило к ситуациям, когда кластеризация далее невозможна.

Несмотря на это, алгоритм проводит кластеризацию. Устранение данных недостатков может расширить границы его применимости.

## Список литературы

**[Варшавский и др., 1984]** Варшавский В.И., Поспелов Д.А. Оркестр играет без дирижера: размышления об эволюции технических систем и управления ими. – М.:Наука, 1984, –208с.

**[Каляев и др., 2009**] Каляев И.А., Гайдук А.Р., Капустян С.Г. Модели и алгоритмы коллективного управления в группах роботов /. – М.: Физматлит, 2009.

**[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" (20-22 мая 2013) Сб. научных трудов. В 3-т., М: Физматлит, 2013, Т.2, с.730-739

**[Карпов и др., 2015]** Карпов В.Э., Карпова И.П., Leader election algorithms for static

swarms //Biologically Inspired Cognitive Architectures №12, 2015, с.54-64

**[Карпов, 2016]** Карпов В.Э., Модели социального поведения в групповой робототехнике. //Управление большими системами, М: ИПУ РАН, 2016, Выпуск 59, с.165-232, -388с.

**[Кокорев, 2005]** Кокорев Н., Чернов Б., Роение медоносных пчел и противороевые приемы. – М.: ТИД Континент-Пресс, Континеталь-Книга, 2005. – 48с.

**[Тамбовцев и др., 2004]** Тамбовцев К.А., Салагаев К.А., Пырялин Г.Л., Яковлева М.П., Ишмуратов Г.Ю. Особенности применения препарата «Апирой» // Пчеловодство. 2004. No 3. С. 13-14.

**[Цетлин, 1969]** Цетлин М.Л., Исследования по теории автоматов и моделированию биологических систем. М.:Наука,1969. 316с.

**[Bina, 2013]** Bina D., Effective cooperation and scalability in multi-robot teams for automatic patrolling of infrastructures, 2013.

**[Caro et al.. 2012]** Di Caro G.A., Ducatelle F., Gambardella L., A fully distributed communication-based approach for spatial clustering in robotic swarms, in: Proceedings of the Second Autonomous Robots and Multi-Robot Systems Workshop (ARMS), Affiliated With the 11th International Conference on Autonomous Agents and Multi-agent Systems (AAMAS), Valencia, Spain, 2012,

[**Chen, et al, 2012]** Segregation in swarms of e-puck robots based on the Brazil nut effect, in: Proceedings of the 11th International Conference on Autonomous Agents and Multi-Agent Systems, vol. 1., International Foundation for Autonomous Agents and Multi-Agent Systems, Richland, SC, 2012, pp. 163–170.

**[Cruz et al., 2016]** Cruz, N.B., Nedjah, N., de Macedo Mourelle, L., 2016. Robust distributed spatial clustering for swarm robotic based systems. *Applied Soft Computing*, p.11.

**[Gross et al., 2009]** Gross R., Magnenat S., Mondada F., Segregation in swarms of mobile robots based on the brazil nut effect, in: Proceedings of International Conference on Intelligent Robots and Systems – IROS 2009, IEEE/RSJ, 2009, pp. 4349–4356

**[Portugal et al., 2013]** Portugal D., Rocha R., Multi-robot patrolling algorithms: examining performance and scalability. Advanced Robotics, 27(5), pp.325–336.

**[Rubenstein et al., 2010]** Rubenstein M., Nagpal R., Kilobot: A Robotic Module for Demonstrating Behaviors in a Large Scale (2^10 Units) Collective. Proceedings of the IEEE 2010 International Conference on Robotics and Automation Workshop, Modular Robotics: State of the Art, (D), pp.47–51.

**[Tan et al., 2013]** Tan Y., Zheng Z., Research Advance in Swarm Robotics. Defence Technology, 9(1), pp.18–39.

**[Yogeswaran et al., 2010]** Yogeswaran M., Ponnambalam S.G., Swarm Robotics: An Extensive Research Review. In Advanced Knowledge Application in Practice, In Tech. pp. 259–278.

**УДК 004.896**

# МОДЕЛИ СТАЙНОГО ПОВЕДЕНИЯ РОБОТОВ[1]

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

**Аннотация.** Предложены модели стайного поведения агентов (роботов) на основе критериев возможности достижения цели и их взаимной полезности при выполнении совместных действий. Рассмотрены стайные алгоритмы «ленивых» и «эгоистичных» агентов.

**Ключевые слова:** Роботы, агенты, стайный алгоритм, кооперация.

## Введение

Идеология командной работы агентов позволяет с помощью множества несложных технических устройств (роботов), выполнять сложные задания в результате кооперации их возможностей. В настоящее время решение этой задачи исследуется в рамках теории группового управления роботами [Каляев и др. 2009], а также в теории мультиагентных систем и искусственного интеллекта. В теории мультиагентных систем и искусственного интеллекта исследуются команды интеллектуальных роботов, с так называемой ментальной архитектурой. В этой работе такие подходы не рассматриваются.

В теории группового управления интерес представляют роевые и стайные подходы к управлению группой роботов. Роевые алгоритмы, основанные на локальном взаимодействии множества однородных роботов (роя роботов), обеспечивают их скоординированное движение к цели, обход препятствий и т.д. [Павловский и др., 2015]. Стайные алгоритмы предполагают взаимодействие функционально разнородных роботов на основе их общих знаний о состоянии среды функционирования и индивидуальных правил поведения агентов в стае.

В живой природе существует множество примеров стайного поведения насекомых и животных [Карпов, 2016]. Считается, что такое поведение является оптимальным. Поэтому, в качестве одной из возможностей повышения качества командной работы агентов (роботов) является

[1] Работа выполнена при поддержке ОФИ_м (проект № 16-29-04412) и РФФИ (проект № 15-01-07900)

моделирование поведения стайных насекомых или животных в алгоритмах командной работы искусственных агентов. В этой работе, на основе исследований социальных психологов предложены модели стайного поведения агентов в процессах образования и функционирования команд агентов.

## 1. Стайное и коллективное поведение агентов (роботов)

Теоретические вопросы создания команд технических устройств (роботов) с системных позиций были рассмотрены в работе [Каляев, и др., 2009]. Здесь группа роботов $R_i$, $i$=1,…$N$, каждый из которых характеризуется вектором состояний R=($r_1$,…,$r_n$) (вектором значений его характеристик) помещен в среду E, которая также характеризуется вектором состояния, E=(e$_1$,…,$e_n$). Пара векторов $S$=⟨$R$, $E$⟩ характеризует состояние системы «группа роботов – среда». Каждый робот может выполнять действия $A$, изменяющие состояние системы «группа роботов – среда». Непредвиденные изменения среды характеризуются вектором $g(t)$. Действия роботов направлены на то, чтобы перевести текущее состояние системы $S_c$ в некоторое желаемое состояние$S_c^f$. Изменение состояния системы «группа роботов – среда», как результата совместных действий роботов описывается системой дифференциальных уравнений. Ставится задача определения совместных действий роботов, позволяющих достичь поставленной цели $S_c^f$. Оценка действий всех роботов для достижения цели осуществляется с помощью оценочного функционала (1).

$$Y_c = \Phi(S_c^f, t_f) + \int_{t_c}^{t_f} F(S_c(t), A_c(t), g(t), t)d(t) \qquad (1)$$

В выражении (1) первое слагаемое характеризует полезность целевого состояния, а второе качество группового управления. Задача взаимодействия группы роботов как команды здесь определена как задача группового управления (управления группой роботов), для решения которой находятся векторы действий всех роботов $A_c(t)$ на промежутке времени [$t_c$, $t_f$], которые обеспечивали бы экстремум функционала (1).

В работе рассматривается два подхода к выработке группового управления: централизованный и децентрализованный. При централизованном управлении векторы действий всех агентов вычисляются в едином центре до решения ими задачи. При децентрализованном управлении роботы обладают автономностью, способны обмениваться информацией и координировать свои действия для достижения индивидуальных целей и общей цели. Это позволяет получить свойства независимости команды от центра, что повышает ее устойчивость к его отказам и отказам отдельных агентов.

Возможны два вида децентрализованного управления: стайное и

коллективное. Как первый, так и второй вид децентрализованного управления предполагает командное поведение роботов для достижения общей цели. Однако, при стайном управлении агенты (роботы) не обмениваются между собой информацией, а строят свою стратегию достижения цели на основе анализа текущего состояния среды функционирования. При коллективном управлении, агенты по каналу связи договариваются о совместном достижении поставленной цели.

В классической постановке стайного поведения каждый робот пытается достичь экстремума функционала (1) самостоятельно, анализируя состояние среды функционирования [Каляев, и др., 2009].

Алгоритм действий роботов заключается в следующем. Роботы анализируют состояние среды функционирования, и выполняют действие для достижения цели, изменяя состояние среды. При этом они не знают, какие действия предприняли другие роботы. Поэтому, на следующем шаге роботы вновь анализируют состояние среды функционирование, которое уже отражает результат действия всех роботов, и каждый из них принимает новые действия для достижения цели. Команда из роботов образуется, если у них есть общая цель. Это необходимое, но не достаточное условие является критерием образования команды в стайном алгоритме. В стайном алгоритме роботы не знают, что они команда, сколько роботов в команде и какие у каждого из роботов возможности для достижения общей цели. В этом случае говорить об оптимальности действий всей команды не приходится, поскольку нет возможности координации их совместных действий. Поведение такой команды будет напоминать суетливое поведение толпы. Действительно, если в команде из множества роботов, есть один или два робота, которые самостоятельно или, объединив усилия способны достичь цели, то остальные члены команды могут «отдыхать», не растрачивая свои ресурсы.

Повысить качество командной работы стаи роботов можно, если кроме информации о состоянии среды функционирования станут общедоступными сведения о целях роботов и имеющихся у них ресурсах для достижения цели. Далее рассмотрим несколько стайных алгоритмов, позволяющих повысить качество их работы.

## 2. Стайное поведение агентов (постановка задачи)

Общая постановка стайной работы команды роботов следующая. Рассматривается множество агентов (роботов) $A=\{a_i\}$, обладающих свойствами (параметрами) $F=\{f_i\}$. Для каждого свойства каждого из объектов (агентов) этой среды определено упорядоченное множество их возможных значений, $Z=\{Z_i\}$, где $Z_i=\{z_{i1},...z_{iq}\}$, $z_{iq+1} \succ z_{iq}$, $q=0...n$-1.

Среда функционирования агентов определяется как прямые

произведение множеств значений всех свойств агентов, $SF = \times_i Z_i$.

Вектор значений всех агентов $Y(t) = (z_{1j}, \ldots, z_{nb})$, $z_{ij} \in Z_i$, $\forall i$, определяет состояние среды функционирования.

Динамика изменения состояния среды функционирования происходит в случаях изменения агентами своих свойств и представляется как отображение:

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

где $W$ – система правил, заданных на множестве всех возможных состояний среды $W: \times_i Z_i \rightarrow \times_i Z_i$; $Y(t)$, $Y(t+1)$ – состояния среды в последовательные моменты времени.

Отметим, что отображение $W$ определяет закономерности среды функционирования, на основе которых агенты принимают решения о своих действиях по достижению цели.

Каждый агент характеризуется следующим кортежем:

$$\langle g_q, r_q, \mu_q(Y_q, g_q) \rangle,$$

где

1) $g_q = (z_{1j}^{g}, \ldots, z_{nb}^{g})$ – вектор целевых значений агента $q$, где $g_q \in SF$;
2) $r_q = (z_{1j}^{r}, \ldots, z_{nb}^{r})$ – стратегия достижения цели агента $q$, где $r_q \in U_q$, $U_q = \times_i Z_i^r$, $Z_i^r \subseteq Z_i$ – ресурсы агента $q$.

Считается, что агент $q$ применяет стратегию $r_q$ для достижения своей цели $g_q$, предполагая, что другие агенты никаких действий не совершали. Прогноз изменения состояния среды на $n$ шагах определится из соотношения (1) при условии, что $Y_q(0) = r_q$, т.е. $Y_q(1) = W \circ r_q$, $Y_q(2) = W \circ Y_q(1)$, …, $Y_q(n) = W \circ Y_q(n-1)$.

3) $\mu_q(Y_q(n), g_q)$ – возможность достижения агентом $q$ целевого состояния за счет собственных ресурсов в условиях противодействия других агентов.

При определении возможности достижения целевого состояния считается, что в пространстве состояний $(\times_i Z_i)$ определена метрика $\rho(a, b)$, $a, b \in \times_i Z_i$. Тогда возможность достижения цели агентом определяется как близость прогнозной $Y_q(n)$ и его целевой ситуации $g_q$:

$$\mu_q(Y_q(n), g_q) = \rho(Y_q(n), g_q).$$

По сути, этот показатель определяет потенциальную «силу» каждого агента команды без поддержки других членов команды при условии противодействия противников [Кулинич, 2014].

В этой модели считается, что состояние среды функционирования $Y(t)$, цель каждого агента $g_q$, его ресурсы $r_q$ и возможность достижения цели самостоятельно $\mu_q(Y_q(n), g_q)$ известны всем агентам.

В такой общей постановке задачи необходимым условием командной работы агентов будем считать критерий близости целей агентов. В команду $K$ в стайном алгоритме включаются агенты $a_i$, цели которых близки, т.е. $\forall a_i \in K,\ K \subseteq A,\ \rho(g_i, g_q) \le \varepsilon,\ \forall i, q \in K,\ \varepsilon$ – критерий близости целей. При этом остальные агенты $A \backslash K$ считаются противниками.

Этот критерий, также как и аналогичный критерий в классической постановке задает необходимые условия командной работы.

**Задача заключается** в том, чтобы определить стратегии (действия) $r_i$ всех агентов - $a_i$, включенных в команду $K$, $\forall a_i \in K$, которые позволяют им наилучшим образом достичь общей цели $g$.

В этой постановке задачи мы пока четко не определили критерий, по которому будет определена лучшая совместная стратегия достижения цели множеством агентов. Очевидно, что составляющими этого критерия являются время, общие затраты ресурсов для достижения цели и др.

Для решения этой задачи предлагается ряд алгоритмов поведения агентов в команде. Различные алгоритмы стайной работы агентов будем связывать с количеством и качеством общедоступной информации (уровнем информированности агентов), которой владеют агенты команды.

Вначале рассмотрим классический стайный алгоритм.

## 2.1. Классический стайный алгоритм.

Классический стайный алгоритм в общей постановке описан в терминах дифференциальных уравнений в работе [Каляев и др. 2008]. Далее не изменяя смысла постановки, будем рассуждать о поведении каждого агента терминах конечно-разностных уравнений (3), опираясь на ранее введенные определения.

$$\begin{aligned} &Y_1(t+1) = W\,(Y(t) + r_{1q}) \\ &\ldots\ldots\ldots\ldots\ldots\ldots \\ &Y_n(t+1) = W\,(Y(t) + r_{nq}) \end{aligned} \qquad (3)$$

Каждый агент на основе знаний о закономерностях среды функционирования $W$ и о состоянии $Y(t)$ принимает решение о действии $r_{iq}$ из множества доступных ему действий, т.е. $r_{iq} \in U_{iq}$, $U_{iq} = \times_i Z_i^r$, $Z_i^r \subseteq Z_i$.

Действие должно приблизить прогнозируемое им состояния среды $Y_i(t+1)$ к общей цели $g = (z_{1j}, \ldots, z_{nb})$, т.е. критерий выбора действия каждого агента следующий: $\forall a_i \in K$, $\rho(Y_i(t+1), g) < \rho(Y_i(t), g)$.

Особенность стайной работы агентов заключается в том, прогнозируемое состояние среды функционирования каждого агента ($Y_i(t+1)$) отличается от состояния $Y^*(t+1)$, полученного в результате совместных действий агентов. Это означает, что на следующем шаге

работы алгоритма в систему уравнений (3) вместо состояния среды $Y(t)$ подставляется $Y^*(t+1)$. И так далее на всех $n$ шагах реализации алгоритма.

Работа алгоритма останавливается, если на очередном шаге $n$, совместными усилиями агентов цель будет достигнута, т.е. $\rho(Y^*(t+n), g)<\varepsilon$.

В приведенном алгоритме все агенты действуют одновременно, не согласовывая свои действия, что приводит к неоптимальному результату.

### 2.2. Итерационный стайный алгоритм

В работе [Каляев и др., 2008] приводится итерационный алгоритм согласования действий всех роботов (агентов), который предполагает наличие между ними канала обмена информацией. Согласно этому алгоритму агенты упорядочены, например, по номерам и выбирают свои действия последовательно в порядке возрастания номеров. Причем первый, выбрав действие, сообщает о нем второму, который выбирает свое действие, учитывая действие первого, и сообщает об этом третьему агенту и т.д. Такая последовательная передача информации о действиях от роботов с меньшими номерами всем последующим роботам позволяет согласовать их действия некоторым оптимальным образом.

В этом алгоритме, до начала решения некоторой задачи группой роботов на основе последовательного обмена информацией о своих действиях формируется общий план последовательных действий агентов достижения общей цели. Далее, считается, что реализация этого плана позволяет оптимальным образом решить поставленную задачу.

Однако, необходимость скоординированного обмена информацией между роботами не позволяет говорить, что такой алгоритм относится к стайным алгоритмам, в которых агенты выбирают действия на основе анализа состояния среды функционирования и по косвенным признакам.

Такой алгоритм не применим, если среда функционирования динамичная, есть агенты–противники, препятствующие достижению цели команды агентов, и команда агентов будет вынуждена, при каждой смене состояния среды функционирования строить новый план оптимального взаимодействия агентов. Построение общего плана может потребовать много времени, что в динамических ситуациях неприемлемо.

Далее будут рассмотрены стайные алгоритмы, в которых применяется широковещательный канал обмена информацией (доска объявлений), не предполагающий скоординированного диалога агентов.

## 3. Стайный алгоритм на основе критериев возможности достижения цели и взаимной полезности агентов

Этот алгоритм основан на анализе возможности достижения цели каждым агентом самостоятельно и их взаимной полезности. Вначале

формально определим взаимную полезность агентов [Кулинич, 2014].

Агенты $q$ и $i$ называются взаимно полезными, если объединение их стратегий $r_i \oplus r_q$ увеличивает возможность достижения собственных целей ($g_q$, $g_i$) агентами $q$ и $i$ в условиях противодействия их противников. То есть если $\mu(Y_{i+q}, g_i) < \mu(Y_i, g_i)$, $\mu(Y_{i+q}, g_q) < \mu(Y_q, g_q)$.Степень полезности $P(i, q)$ агента $q$ для агента $i$ определяется из соотношения [Кулинич, 2014]:

$$P(i,q) = 1 - \frac{\mu(Y_{i+q}, g_i)}{\mu(Y_i, g_i)} \qquad (4)$$

где $\mu(Y_{i+q}, g_i)$, – возможности достижения агентами $q$ и $i$ своих целей при объединении их стратегий; $\mu(Y_q, g_q)$, $\mu(Y_i, g_i)$ – возможности достижения агентами $q$ и $i$ своих целей собственными силами в условиях противодействия противников.

Степень полезности следующего агента, для команды из двух агентов $q$ и $i$ будем определять из соотношения:

$$P((i,q), j) = 1 - \frac{\mu(Y_{((i+q)+j)}, g_i)}{\mu(Y_{((i+q))}, g_i)} \qquad (5)$$

Таким образом, для определения полезности вновь добавляемого агента к существующей команде, команда считается единым агентом.

Считается, что каждый агент или команда на очередном такте работы помещает на доску объявлений информацию: о действиях $r_{iq}$ (стратегию достижения цели) и о возможности достижения цели самостоятельно - $\mu_q(Y_q(n), g_q)$. Эти сведения доступны всем агентам.

Эту информацию каждый агент анализирует автономно по одинаковым алгоритмам. Поэтому результаты вычислений у всех агентов будут одинаковыми, что исключает необходимость их согласования.

Алгоритм функционирования агентов следующий.

Каждый агент упорядочивает всех агентов (или команду агентов) в порядке возрастания их возможностей достижения цели.

1. Если в перечне агентов есть агент (команда), способные достичь цели самостоятельно, то агент (команда) и будут решать задачу достижения цели, т.е. $\exists i | \mu(Y_i, g_i) < \varepsilon$. Остальные агенты в этом процессе не участвуют.
2. Если такого агента нет, то в качестве члена команды выбирается агент с большей возможностью достижения цели, для которого по формулам (4,5) подбирается агент с наибольшей взаимной полезностью. Эта команда будет работать на следующем такте работы стайного алгоритма. Остальные члены команды бездействуют.

Особенность стайного алгоритма заключается в том в том, что на каждом шаге работы алгоритма изменяются возможности достижения цели и, соответственно, взаимные полезности агентов.

3. Шаги алгоритма 1,2 и 3 повторяются до достижения цели, т.е. $\rho(Y^*(t+n), g)<\varepsilon$, для новых значений возможности достижения цели агента и их взаимной полезности.

Таким образом, на каждом шаге этого алгоритма состав команды может меняться по количеству агентов или составу. Но важно, то, что на каждом шаге команда всегда будет содержать агента (команду) с наибольшей возможность достижения цели и самого взаимно полезного агента. Считается, что такой подбор команды позволит повысить качество командной работы агентов и приблизить их поведение к оптимальному.

Далее рассмотрим стайные алгоритмы, в которых алгоритмы командной работы стаи искусственных агентов (роботов) строятся на основе моделирования поведения людей в социуме.

## 3.1 Стайный алгоритм «ленивых» агентов

Этот алгоритм будем формулировать в терминах ранее введенной постановки задачи. Общее описание алгоритма следующее.

4. Каждый агент упорядочивает всех потенциальных членов команды по возможности достижения ими цели.
5. Если есть агент, способный достичь цели самостоятельно, то он достигает ее. Остальные агенты бездействуют.
6. Если такого агента нет, (здесь начинаются отличия от ранее описанного алгоритма), то команда принимает решение, чтобы лучший агент попытался самостоятельно достичь целевого состояния.
7. Остальные агенты, наблюдая за состоянием среды функционирования, ждут, когда этот агент исчерпает свои возможности достижения цели. Т.е. множество агентов находятся в состоянии «ленивого» ожидания результатов работы лучшего агента.
8. Если, наблюдаемый агент исчерпал свои возможности, то из множества «ленивых» агентов выбирается агент с большей возможность достижения цели, который пытается ее достичь.
9. Множество «ленивых» агентов ждет, когда и этот агент исчерпает свои возможности и назначает нового исполнителя.
10. Процесс заканчивается, когда цель будет достигнута, т.е. если $\rho(Y^*(t+n), g)<\varepsilon$.

Этот алгоритм похож на итерационный алгоритм [Каляев и др., 2009], но его отличие заключается в том, что выбор очередного агента для

включения его в команду происходит в процессе решения агентами задачи, а не до его начала.

## 3.2. Стайный алгоритм «эгоистичных» агентов

Этот алгоритм предполагает, что устойчивая команда многократно выполняет некоторую работу. Например, в социальных системах, в бригаде рабочих при многократном выполнении проектов и т.д. между членами коллектива могут сложиться отношения (симпатии, неприязни, лидерства и т.д.), которые могут сказаться на работе коллектива. Отношения между агентами, возникающие в коллективе связаны с их индивидуальными отличиями и благами, которые могут получать агенты с большими возможностями. Например, в ранее рассмотренных алгоритмах, основанных на возможности достижения цели и взаимной полезности, основную командную работу будут выполнять только агенты с лучшими возможностями. Остальные будут «отдыхать». Если лучшие агенты за свою работу будут получать больш*и*е блага, то остальные («ленивые») агенты могут стремиться получать эти блага, перехватывая работу у лучших агентов, даже в том случае если качество работы всей команды ухудшиться. Таких агентов будем называть эгоистичными.

В алгоритме «эгоистичных» агентов каждый агент кроме характеристики возможности достижения цели самостоятельно имеет еще и характеристику его работы $O_i$ в предыдущих циклах работы, величина которой пропорциональна объему выполненной им работы и, соответственно, благам, которые агент за эту работу получил. Уровень эгоистичности агентов может быть определен из соотношения:

$$D_{ij} = \frac{|O_i - O_j|}{O_i + O_j}$$

В этом соотношении эгоистичность агентов $D_{ij}$=0, если они выполнили равную по объему работу, и соответственно, равна 1, если один из них никакой работы не выполнял – «отдыхал» («ленился»).

В алгоритме «эгоистичных» агентов считается, что сведения о выполненной работе каждого агента является общедоступными и, следовательно, агенты могут быть упорядочены по этой характеристике.

Рассмотрим алгоритм «эгоистичных» агентов.

1. Каждый агент упорядочивает всех потенциальных членов команды в два упорядоченных множества: по возможности достижения ими цели и по уровню эгоистичности агента относительно агента с максимальным объемом, выполненной работы.
2. Если есть агент, способный достичь цели самостоятельно, и его эгоистичность максимальна (или эгоистичности всех агентов равны), то он достигает ее. Остальные агенты бездействуют.

3. Если такого агента нет, то команда принимает решение, чтобы агент с максимальным уровнем эгоистичности попытался самостоятельно достичь целевого состояния.
4. Процесс заканчивается, когда цель будет достигнута, т.е. если $\rho(Y^*(t+n), g)<\varepsilon$.

Этот алгоритм позволяет равномерно распределить работу между всеми членами команды и обеспечить им одинаковые вознаграждения за работу в команде.

## Заключение

В работе проанализирован классический стайный алгоритм роботов, выявлены его недостатки при формировании общего плана решения задачи в динамических ситуациях. Предложены стайные алгоритмы на основе критериев возможности самостоятельного достижения агентом цели и взаимной полезности агентов. На основе этих же критериев рассмотрены стайные алгоритмы «ленивых» и «эгоистичных» агентов.

Дальнейшие имитационные исследования формальных стайных алгоритмов позволят предложить их для реализации в командах роботов.

## Список литературы

[**Каляев и др., 2009**] Каляев И.А., Гайдук А.Р., Капустян С.Г. Модели и алгоритмы коллективного управления в группах роботов / М.: ФИЗМАТЛИТ, 2009.-279 с.

[**Павловский, 2015**] Павловский В.Е., Павловский В.В. Математическая модель двумерной гомогенной стаи роботов// Искусственный интеллект и принятие решений. М.: URSS, 2015, №4, с. 62-71.

[**Карпов, 2016**] Карпов В. Э. Модели социального поведения в групповой робототехнике / Управление большими системами. Выпуск 59. М.: ИПУ РАН, 2016. С.165-232.

**[Кулинич, 2014]** Кулинич А. А. Модель командного поведения агентов (роботов): когнитивный подход / Управление большими системами. Выпуск 51. М.: ИПУ РАН, 2014. С.174-196.

**УДК 004.896**

# АЛГОРИТМ ПЕРЕСТРОЕНИЙ ГРУППЫ БЕСПИЛОТНЫХ ЛЕТАТЕЛЬНЫХ АППАРАТОВ С ИСПОЛЬЗОВАНИЕМ АВТОПИЛОТА PIXHAWK

Н.А. Михайлов (*mikhailov.mai@gmail.com*)
Московский авиационный институт (национальный исследовательский университет) (МАИ)

**Аннотация.** В статье рассматривается практическая реализация алгоритмов управления автономным беспилотным летательным аппаратом в составе группы с использованием полетного контроллера Pixhawk. Рассматривается задача согласованного перестроения группы по заданным формациям. Каждый участник группы находится под управлением автопилота, который получает команды от бортового компьютера, на борту которого установлено программное обеспечение, реализующее алгоритмы управления. В работе представлена схема построения системы управления БПЛА. Проведено компьютерное моделирование. Результатом моделирования стали графики, демонстрирующие работоспособность алгоритмов;

**Ключевые слова:** система управления, беспилотный летательный аппарат, групповое применение.

## Введение

В последние годы все большее распространение получают задачи по разработке систем управления (СУ) автономными беспилотными летательными аппаратами (БЛА) [Mo 2011], [Moon 2009]. Интерес к данной проблеме объясняется преимуществами данных систем:

- Возможность решать целевые задачи в опасных для человека зонах (зоны радиоактивной опасности, зоны лесных пожаров, зоны военных конфликтов и т.п.);
- Способность совершать полеты на большие расстояния (за пределами зон ручного управления);
- Отсутствие человеческого фактора.

Возрастающая сложность и объемы решаемых современных задач вынуждает разрабатывать СУ БПЛА, согласованно функционирующих в составе группы. Этот факт объясняется основными преимуществами такого подхода:

- Увеличение вероятности успешного выполнения задачи группой;
- Увеличение производительности за счет возможности одновременного обследования территории, с последующей оптимизацией маршрута полета, на основе имеющихся данных с другого члена группы;
- Увеличение производительности за счет возможности постановки различных задач для разных участников группы.

Известно большое количество работ, связанных с решением задачи согласованного управления группой БЛА [Waydo 2006], [Valenti 2004], [Dong 2009], [Crowther 2003]. В то же время ряд задач, например, таких как пространственные перестроения БЛА в группе, реализация алгоритмов управления на конкретных полетных контроллерах и др., требует дополнительных решений.

В данной работе предлагается алгоритм автоматического согласованного перестроения БПЛА в группе за минимальное время и его реализация на бортовом микроконтроллере и автопилоте (полетном контроллере) Pixhawk.

# 1 Полетный контролер Pixhawk

Полетный контроллер Pixhawk – это продвинутая система автопилотирования, разработанная PX4 (проект отрытых аппаратных средств) и произведенная компанией 3D Robotics. [Pixhawk 3D Robotics] Спецификация автопилота:

Микропроцессор:

- 32-bit STM32F427 Cortex M4 ядро с FPU;
- 168 MHz/256 KB RAM/2 MB флэш-памяти;
- 32 bit STM32F103 безотказный сопроцессор.

Датчики:

- ST Micro L3GD20 3-axis 16-bit гироскоп;
- ST Micro LSM303D 3-axis 14-bit акселерометр / магнитометр;
- Invensense MPU 6000 3-axis акселерометр/гироскоп;
- MEAS MS5611 барометр.

В качестве прошивки автопилота может выступать одна из следующих:

1. PX4 [PX4,a];
2. APM (ArduPilot Mega) [Ardupilot Mega,a].

Оба проекта прошивки автопилота имеют открытый код, что делает их привлекательными для разработчиков алгоритмов управления БЛА. [PX4,b], [ Ardupilot Mega,b].

В данной работе рассматривается прошивка APM (Arducopter 3.3). Данная прошивка позволяет провести имитационное моделирование поведения автопилота Pixhawk с использованием в качестве модели

аппарата встроенные уравнения (SITL режим работы автопилота [SITL]) или внешние (HIL режим работы автопилота [HIL]).

В работе исследуется режим SITL. В этом режиме компьютерная модель автопилота взаимодействует по протоколу TCP/IP с компьютерной моделью БПЛА. В изначальном варианте компьютерная модель БПЛА заложена в исходный код прошивки. При необходимости эту модель можно корректировать.

Математическая модель регуляторов углового и линейного положения представляет собой ПИД регуляторы, коэффициенты которых можно настроить.

Общение с автопилотом происходит с использованием протокола MAVLink [MAVLink].

## 2 Математическая модель группы

Для математического описания группы разобьем пространство в горизонтальной плоскости на сетку размером NxN, где N – количество аппаратов в группе (см. Рис. 1). Размер сетки соответствует требуемым значениям интервалов и дистанций между аппаратами. Тогда положение каждого аппарата в сетке будет определяться двумя числами от 1 до N-1.

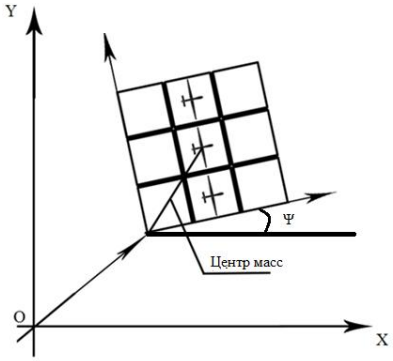



Рис. 1. Математическая модель строя.

При таком описании строя важными характеристиками так же будут центр масс строя и курс. Эти две характеристики необходимы для выработки такого управления, чтобы совершать групповой маневр не нарушая строя. Т.е. для того, чтобы развернуть строй на требуемый угол курса необходимо рассчитать положения аппаратов относительно центра масс соответствующие этому углу, а после развернуть сами аппараты в том же направлении.

В дальнейшем построение группы в пространстве можно выполнять относительно центра масс формации. Это удобно для предполетного планирования, в качестве промежуточного пункта маршрута (ППМ) задавать центр масс формации (см. Рис. 2).

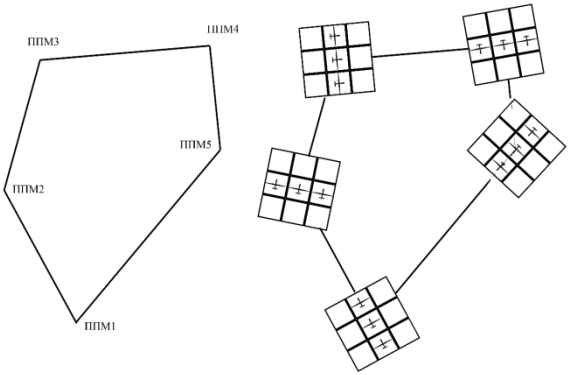



Рис. 2. Вариант предполетного задания.

На рис. 3. показаны возможные варианты построения

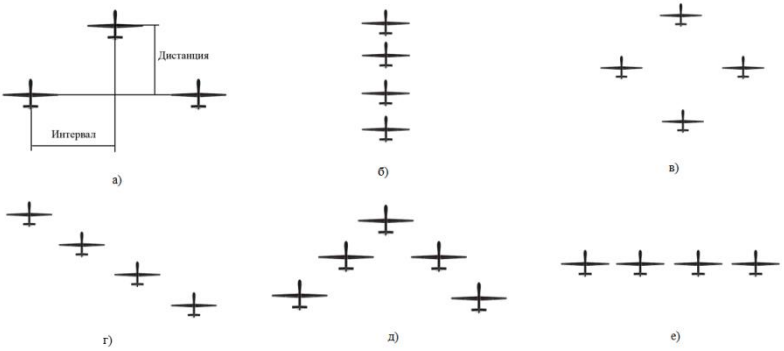



Рис. 3. а) Дистанция и интервал; б) «Колонна»; в) «Ромб»; г) «Пеленг»; д) «Клин»; е) «Фронт»

## 3 Алгоритм перестроения

Пусть имеется набор штатных «формаций» - пространственных построений БПЛА: «строй», «колонна», «ромб», «клин», «пеленг». Расстояния между отдельными БПЛА (интервал и дистанция) для определенных условий заданы.

Необходимо за минимальное время перестроится из исходной в требуемую формацию. При этом безопасность перестроений БПЛА, связанная с вероятностью столкновений, задается допустимым сближением (расстоянием между БПЛА).

Вариантом перестроения является набор частных траекторий (переходов) полета каждого БПЛА от исходной позиции в текущей формации к некоторой конкретной позиции в требуемой формации.

Суть перестроения заключается в определении положения БПЛА в новой формации. Для этого необходимо совместить начало координатной сетки одной формации с другой. Таким образом каждый БПЛА в новой формации может занимать N различных положений. Но не все они целесообразны, поэтому необходимо использовать некоторый алгоритм распределения, который нашел бы такую совокупность положений в новой формации («БЛА-Цель»), при которой суммарная длина переходов была минимальна. Данная задача известна в комбинаторике как «Линейная задача о назначениях» [Асанов 2001], так как количество БПЛА равно количеству мест в новой формации.

Одним из классических подходов решения задачи о назначениях является «Венгерский алгоритм». Он позволяет решить задачу за полиноминальное время

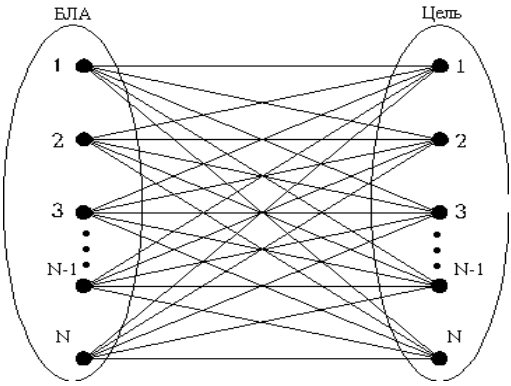



Рис. 3. Полный двудольный граф переходов.

Таким образом можно сформировать базу данных (БД) по всем возможным перестроениям варьируя значения интервалов, дистанций, количества БПЛА и форму построений

В дальнейшем каждый БПЛА оснащается такой БД. Алгоритм перестроения можно описать следующим образом:

1. Приходит сигнал о необходимости перестроения в некоторую формацию по ключу;

2. Из памяти извлекается текущий ключ формации, а также значения интервалов, дистанции и количества БПЛА в строю $I, D, \mathrm{N}$;
3. По данным параметрам из БД извлекается массив назначений $i_{треб.}, j_{треб.}$;
4. Из памяти извлекается текущий положение в построении $i_{тек.}, j_{тек.}$;
5. С учетом текущего и требуемого положения в формации определяем необходимое смещение $\Delta i, \Delta j$:

$$\Delta i = i_{треб.} - i_{тек.}$$

$$\Delta j = j_{треб.} - j_{тек.}$$

6. Из собственной навигационной системы извлекаем собственное положение $x_{БЛА}, y_{БЛА}$;
7. Определяем требуемое положение относительно рассчитанного в пункте 5 смещения, текущего положения и разворота на угол $\Psi_{форм.}$;

$$\begin{pmatrix} \Delta x_{форм.} \\ \Delta y_{форм.} \end{pmatrix} = A_z\left(\Psi_{форм}\right)\begin{pmatrix} I & D \\ D & I \end{pmatrix} \cdot \begin{pmatrix} \Delta j \\ \Delta i \end{pmatrix} + \begin{pmatrix} x_{\text{БЛА}} \\ y_{\text{БЛА}} \end{pmatrix}, \text{ где}$$

$\Delta x_{форм.}, \Delta y_{форм.}$ – требуемые смещения для БПЛА по оси X и Y стартовой СК соответственно;

$A_z\left(\Psi_{треб}\right)$ – матрица поворота относительно оси Z стартовой системы координат;

$I$, $D$ – заданные значения интервала и дистанции.

8. Отправляем на автопилот рассчитанное значение;
9. Опрос навигационной системы автопилота до достижения требуемого положения. Выработка сигнала о достижении требуемого положения и отправка его по каналу связи.

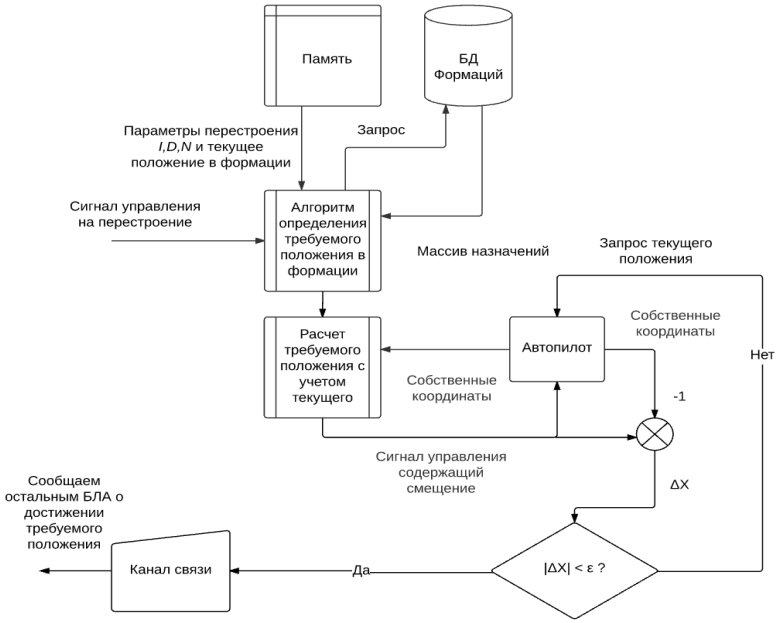



Рис. 4. Блок-схема алгоритма перестроения.

# 4 Компьютерное моделирование работы алгоритмов

Для проверки работоспособности алгоритма, был разработан программно-вычислительный комплекс на базе прошивки автопилота Pixhawk, и проведен эксперимент, моделирующий выполнение группой полетного задания. Программный комплекс можно разделить на три большие части:

1. ПО, отвечающее за моделирование работы автопилота и физической модели БЛА. Данное ПО является частью прошивки автопилота Pixhawk;
2. ПО, отвечающее за работу алгоритмов согласованного управления группой БПЛА. Именно это ПО устанавливается на борт микрокомпьютера и отдает команды на автопилот;
3. ПО, отвечающее за визуализацию работы алгоритмов и системы в целом. В данной работе визуализация осуществляется с помощью инструментов Matlab.

Экспериментальное полетное задание определим следующим образом:

1. Начальное построение «Колонна»;
2. Первая точка ППМ координаты X – 0, Y – 30;
3. В первой точке ППМ сменить формацию на «Фронт»;

4. Вторая точка ППМ координаты X – 30, Y – 30;
5. Во второй точке ППМ сменить формацию на «Клин»;
6. Третья точка ППМ координаты X – 30, Y – 0;

Количество БПЛА в группе N = 3, значение интервалов и дистанций между БПЛА $I = D = 5$.

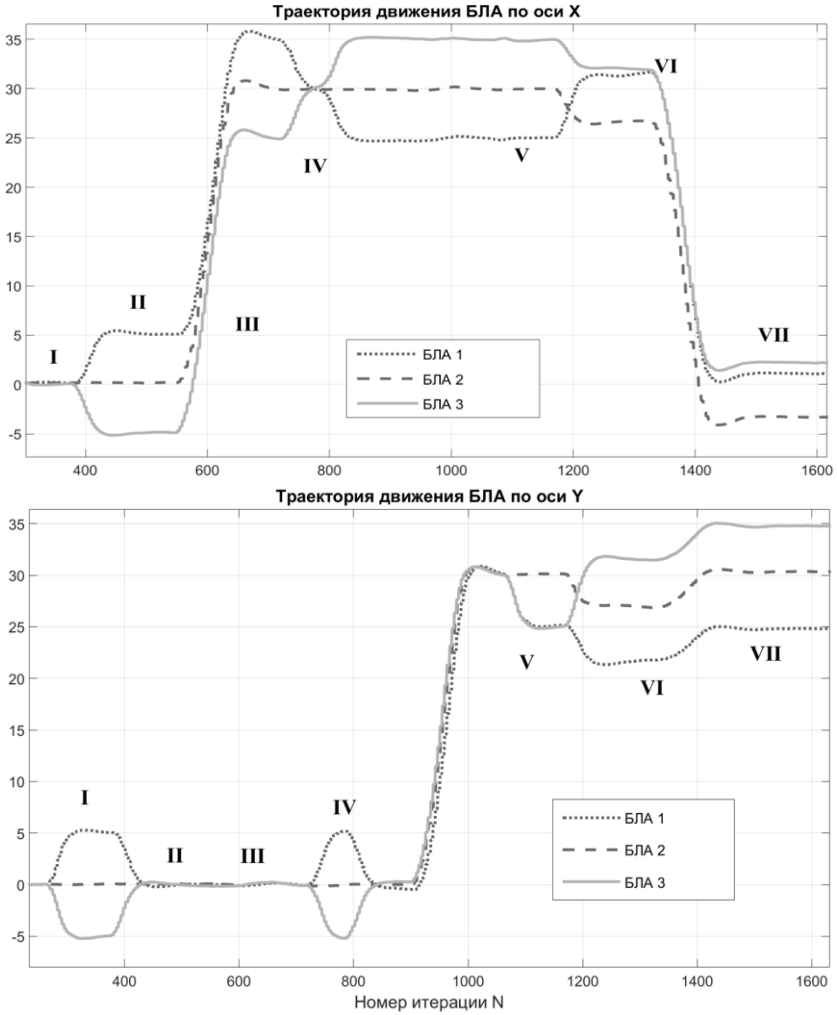



Рис. 5. Траектории движения группы.

Результаты компьютерного моделирования эксперимента представлены на графиках (см. Рис. 5), размерность координат на плоскостях X и Y метры: **I** начальное построение «Колонна», **II** разворот на ППМ1, **III** группа достигла ППМ1 (во время движения аппараты не столкнулись), **IV** смена формации с «Колонна» на «Фронт» и движение к ППМ2, **V** смена формации с «Колонна» на «Клин», **VI** разворот на ППМ3, **VII** движение к ППМ3. По достижению ППМ3 каждый БПЛА занял требуемую позицию, что можно считать успешным завершением миссии.

## Заключение

Новизна представленной работы состоит в том, что здесь рассматривается случай построения системы управления БЛА в составе группы для решения задачи согласованного перестроения с использованием конкретного типа автопилота.

В работе представлен алгоритм согласованного перестроения и разработанный программный комплекс, с помощью которого проверяется работоспособность алгоритма.

По результатам компьютерного моделирования, было показано, что группа БПЛА успешно выполнила задание.

Исходный код программы размещен в открытом доступе [Source 2016].

## Список литературы

**[Mo 2011]** Mo Jamshidi, Jose Gome1, Aldo S. Jaimes B. – «Intelligent control of UAVs for consensus-based and network controlled applications» // Applied and Computational Mathematics, 10(1), 35-64. 2011

**[Moon 2009]** Moon J. – «Mission-based guidance system design for autonomous UAVs» // Georgia Institute of Technology. 2009.

**[Waydo 2007]** Waydo S., Hauser J., Bailey R., Klavins E., Murray R.M. – «UAV as a Reliable Wingman: A Flight Demonstration» // IEEE Transactions on Control Systems Technology; p. 680 – 688. 2007.

**[Valenti 2004]** Valenti M., Schouwenaars T., Kuwata Y., Feron E., How J. and Paunicka J. – «Implementation of a Manned Vehicle - UAV Mission System» // Massachusetts Institute of Technology, Cambridge, MA 02139, The Boeing Company, St. Louis, MO 63166. 2004.

**[Dong 2009]** Xiangxu Dong, Ben M. Chen, Guowei Cai, Hai Lin and Tong H. Lee – «Development of a comprehensive Software system for implementing Cooperative control of multiple Unmanned aerial vehicles» // IEEE International Conference on Control and Automation, p. 1629 – 1634, 2009.

**[Crowther 2003]** Crowther B. – «Flocking of autonomous unmanned air vehicles» // School of Engineering University of Manchester, UK. 2003.

**[Pixhawk 3D Robotics]** Pixhawk 3D Robotics официальная страница магазина https://store.3drobotics.com/

**[PX4,a]** Официальная страница разработчиков прошивки PX4

https://pixhawk.org/start

**[Ardupilot Mega,a]** Официальная страница разработчиков прошивки Ardupilot Mega http://ardupilot.com/

**[PX4,b]** Исходный код прошивки PX4 https://github.com/PX4/Firmware

**[Ardupilot Mega,b]** Исходный код прошивки Ardupilot Mega https://github.com/diydrones/ardupilot

**[SITL]** Описание работы режима SITL прошивки автопилота Ardupilot Mega http://dev.ardupilot.com/wiki/setting-up-sitl-on-windows/

**[HIL]** Описание работы HIL режима прошивки автопилота Ardupilot Mega http://copter.ardupilot.com/wiki/hil-quad/

**[MAVLink]**MAVLink протокол https://github.com/mavlink/mavlink

**[Асанов и др. 2001]** М.О.Асанов, В.А.Баранский, В.В.Расин – «Дискретная математика: графы матроиды, алгоритмы.» - // Ижевск: НИЦ "РХД", 288 стр. 2001.

[**Source 2016**] Исходный код программы https://github.com/MihailovJava/UavControl

**УДК 519.876.2**

# ДЕЦЕНТРАЛИЗОВАННОЕ УПРАВЛЕНИЕ ГРУППОЙ БЕСПИЛОТНЫХ АППАРАТОВ НА ОСНОВЕ ИМИТАЦИИ АГРЕГАТНЫХ СОСТОЯНИЙ ВЕЩЕСТВА

Р.Т. Сиразетдинов (*Rif-kat@inbox.ru*)
С.В. Тихонов (*gilgul_ha-n@mail.ru*)
Казанский национальный исследовательский технический университет им. А.Н. Туполева–КАИ, Казань

**Аннотация**. В докладе рассматривается проблема организации автономного движения децентрализованной группы (стаи) беспилотных аппаратов, как множества материальных точек. Предлагаются алгоритмы распределенного управления, основанные на имитации группой беспилотных аппаратов различных агрегатных состояний вещества. Рассмотрены три «агрегатных состояния» группы: твердое, жидкое и газообразное. Исследованы вопросы устойчивости системы при различных вариантах алгоритмов имитации агрегатного состояния вещества. Затронуты вопросы оптимизации одномерного и плоского движения группы.[1]

**Ключевые слова**: группа беспилотных аппаратов, двухуровневое управление, децентрализованное управление, агрегатные состояния вещества.

## Введение

В настоящее время появилась возможность создавать конструкции достаточно малых размеров, например, квадрокоптеры, которые могут автономно перемещаться в пространстве, выполнять поставленные перед ними интеллектуальные задачи и, при этом, иметь небольшую стоимость. Это позволяет вместо одного большого беспилотного аппарата (БА) использовать несколько малоразмерных и недорогих БА, которые совместно могли бы выполнить те же задачи, и, при этом, имели бы ряд преимуществ, таких как малая стоимость и меньшая уязвимость группы БА как системы, малая заметность для людей, для радаров, способность более оперативно принимать решения и передавать информацию, а также

[1] Работа выполнена за счет средств субсидии, выделенной в рамках государственной поддержки Казанского (Приволжского) федерального университета.

больший радиус действия. Кроме того возникают новые свойства группы БА как системы, позволяющие выполнять и другие виды задач, невыполнимые одним БА. Так, на базе группы малых интеллектуальных БА можно организовать пространственно распределенную интеллектуальную вычислительную систему, способную производить параллельную обработку информации. Суммарная вычислительная мощность такой системы будет складываться из мощностей составляющих ее БА. При этом, при потере одного БА будет происходить его замещение другим, что обеспечит надежность вычислительной системы.

Централизованное управление группой БА с увеличением количества БА встречает все больше трудностей. Наличие центрального управляющего БА, во-первых, делает систему более уязвимой и менее надежной, так как достаточно какого-либо сбоя в работе центрального аппарата, и вся группа теряет управляемость. Кроме того, возрастают требования к процессору, памяти, быстродействию центрального управляющего БА, возрастают требования к надежности и мощности системы связи, обеспечивающей информационный трафик центрального БА. Отсюда следует актуальность проблемы разработки децентрализованного управления большой группой беспилотных аппаратов, когда подается единая команда стае, и каждый БА отрабатывает эту команду самостоятельно [Каляев и др., 2010].

Чаще всего, проектируя систему, представляющую из себя группу БА с децентрализованным управлением, вдохновляются творениями живой природы. Так, например, группа БА может имитировать движение стаи животных или роя насекомых. При этом для обмена информацией можно использовать аналоги чувства кворума, как у бактерий [Bassler, 1999], трофаллаксиса, как у пчел, муравьев или термитов [Schmickl, 2008], а также гормонов [Stamatis, 2009]. Каждый БА, в соответствии с общим заданием и информацией, полученной выбранным способом, реализует, так называемое, ситуационное поведение [Поспелов, 1986]. На аппаратном уровне связь между БА может реализовываться на основе радиосигналов, Wifi и Bluetooth, световых, звуковых и даже химических сигналов.

В КНИТУ-КАИ на кафедре Динамики процессов и управления в течение многих лет ведутся исследования, связанные с проблемой оптимизации и устойчивости систем с распределенными параметрами, математического моделирования и управления сложными, трансформирующимися системами. Данная статья является продолжением этих исследований и связана с математическим моделированием такой сложной, пространственно распределенной, трансформирующейся системы, как большая группа БА.

## 1 Постановка задачи

Рассматривается управляемое движение некоторой группы БА, состоящей из большого числа аппаратов. Предполагается, что все БА идентичные.

На каждый БА со стороны среды накладываются ограничения, например, препятствия, которые БА должен облететь. Также для каждого конкретного БА накладывают ограничения все соседние БА. Это и минимально допустимое расстояние между БА, необходимое для предотвращения столкновений, и «видимость» БА хотя бы одним другим БА, для возможности передачи информации и предотвращения потери БА группой, и т.д. Дополнительные ограничения могут быть переданы группе в задании.

Требуется разработать интеллектуальные методы управления группой БА и взаимодействия между БА. Это значит, что необходимо организовать своевременный обмен актуальной информацией между БА и разработать алгоритмы, которые на основе полученной информации будут переводить систему в требуемое по общему заданию состояние.

## 2 Групповое движение беспилотных аппаратов

В данной статье описывается двухуровневый принцип организации управления группой БА. Верхний уровень предполагает централизованное командование оператором, либо формирование целеуказаний автоматизированной системой, в целом для группы БА. На нижнем уровне каждый БА самостоятельно отрабатывает свои алгоритмы движения на основе заданной общегрупповой цели и взаимного положения относительно ближайших БА.

Основной идеей управления группой БА, предлагаемой нами, является имитация группой БА агрегатного состояния вещества. На нижнем уровне управления алгоритмы управления каждого БА составлены так, чтобы группа в целом для оператора представлялась, либо как твердое тело, либо как жидкость, либо как газ. Тогда команды высокого уровня становятся оператору интуитивно понятными. Он имеет конечный набор высокоуровневых команд для управления группой как «веществом», т.е. некоторым физическим телом, а группа БА самостоятельно отрабатывает эти команды на основе заложенных в них типовых алгоритмов нижнего уровня.

Высокоуровневые команды управления группой БА в виде вещества можно разделить на следующие группы:

- Команды перехода в новое агрегатное состояние (АС) с заданными параметрами.
- Команды перемещения группы БА в заданном направлении в

текущем АС.

- Команды, связанные с непосредственным выполнением целевой задачи, например, аэрофотосъемка местности и т.п.

При реализации группового управления БА целью ставится не моделирование реальных свойств вещества в соответствующем АС, а имитация тех ключевых свойств АС вещества, которые могут быть полезными при управлении группой БА. Рассмотрим эти ключевые свойства.

При «твердом» АС группа БА имеет неизменную структуру, удобно в этом случае использовать «кристаллическую» структуру. Все БА жестко располагаются в заданных узлах относительно друг друга. Каждый БА может немного отклоняться от заданного положения по аналогии с движением атомов в кристаллической решетке. Структура «кристаллической решетки» и степень допустимых отклонений задается на высшем уровне управления. Система управления каждого БА самостоятельно отрабатывает свое заданное положение.

Возможен второй способ построения модели «твердого» АС с помощью не рекурсивной уникальной схемы, заданной некоторым массивом узлов, в которых должны расположиться БА. Данный способ предполагает, что есть некоторая конечная схема расположений БА относительно друг друга.

Группа БА в «жидком» АС имеет следующие ключевые свойства:

- Поддержание постоянной плотности, т.е. концентрации БА в пространстве. Следствием этого свойства является постоянный объем пространства, занимаемый группой БА.
- Стремление группы БА занять структуру с минимальной площадью оболочки, покрывающей группу БА. Это требование воспроизводит такое свойство жидкости как поверхностное натяжение.

Группа БА в виде «жидкости» движется компактно, при этом БА не имеют фиксированных позиций в группе и свободно меняются между собой. Группа БА, обтекая или просачиваясь, обходит препятствия. Одним из ограничивающих условий для состояния «жидкости» является минимально допустимое расстояние между БА. Оно необходимо, чтобы не было соударений БА.

Группа БА в «газообразном» АС имеет такое ключевое свойство, как стремление равномерно занять максимальную область. Предполагается, что на верхнем уровне управления задается конфигурация области, т.е. оболочки, которую должен занять «газ». Если же группа БА в состоянии «газа» не ограничена оболочкой, то вся группа занимает максимально возможную область, при которой она остается управляемой. Одним из ограничивающих условий на максимальный объем является радиус

видимости БА друг друга — БА не могут удаляться на большее расстояние.

## 3 Относительное движение беспилотных аппаратов

Рассмотрим относительное движение аппаратов под действием алгоритмов нижнего уровня. Группа БА имитирует движение молекул вещества под действием внутренних сил. Результирующая этих сил и будет требуемой тягой каждого БА. При этом БА рассчитывает силы, которые действовали бы на него, если бы он был частицей вещества, находит их результирующую и отрабатывает ее действие собственной тягой. Величины действующих сил вычисляются в соответствии с заданным законом движения.

В каждом из представленных нами АС любой БА имитирует действие на него сил со стороны остальных БА, которые можно разделить на силы отталкивания и притяжения. Силы отталкивания необходимы для предотвращения столкновения БА. А силы притяжения для предотвращения потери связи между БА.

Задавать взаимодействие между БА мы можем сами, тем самым меняя свойства группы БА. Для простоты допустим, что в движение одномерное и недемпфированное. Представим БА материальными точками.

Сравним действие различных законов движения. Их можно разделить на 2 класса: линейные (такие как закон Гука) и не линейные (такие как закон Кулона, гиперболический и квадратичный законы и др.).

Основное достоинство нелинейных законов в отличие от линейных: при убывании расстояния между аппаратами, сила, действующая между ними, резко возрастает вплоть до бесконечности. Тем самым мы гарантируем, что БА никогда не столкнутся. Но это возможно только в идеальной системе. В реальности мы не можем приложить к аппарату бесконечную тягу, и отработка заданной программы идет не мгновенно, что накладывает ограничения на начальные условия системы.

Рассмотрим двумерное плоское движение БА. Для этого представим ограниченную внешнюю среду в виде квадрата, внутри которой движутся БА. АС выберем газообразное. На рисунке 1 представлено мгновенное расположение БА внутри ограниченной площади.

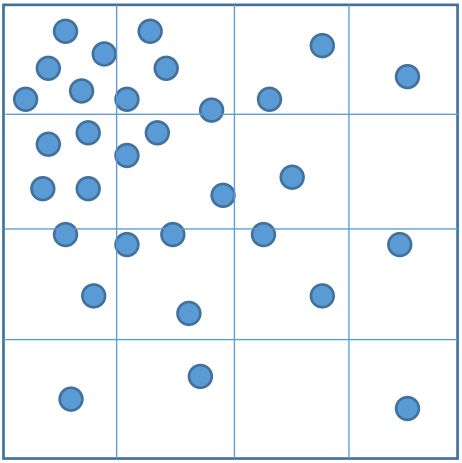


Рис. 1. Мгновенное расположение БА в пределах заданной области.

Оценим плотность распределенной системы. Для этого поделим допустимую область на более мелкие области. Построив двумерную гистограмму, получим мгновенную оценку плотности БА в подобластях. На ней видим, что у нас имеются области с повышенной мгновенной плотностью БА и области с пониженной мгновенной плотностью БА.

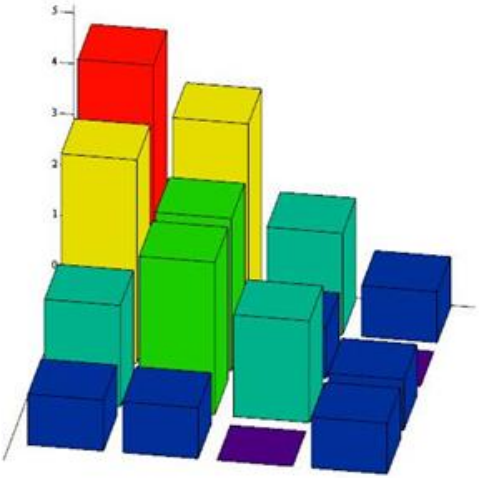



Рис. 2. Гистограмма оценки мгновенной плотности БА в подобластях.

В ходе движения в «газообразном» АС БА должны заполнять допустимую область равномерно, то есть плотность БА в каждой подобласти должна быть примерно равна средней плотности всей группы. Тогда в процессе движения БА должны постепенно рассредоточиться по всей допустимой площади, так чтобы значение плотности в каждой подобласти лежало в допустимых пределах. Пределы задаются разработчиком. Время, за которое значение плотностей в подобластях достигнет допустимых пределов и больше не выйдет за них, и будем считать критерием оптимальности данной системы.

## 4 Заключение

В данной работе предложен подход к организации интеллектуального управления большими группами БА. Он основан на децентрализованном управлении и имитации группой БА агрегатных состояний вещества. Это позволяет управлять большой группой БА с помощью интуитивно понятных оператору высокоуровневых команд. При этом алгоритмы поведения, т.е. управления, каждого БА достаточно автономны и просты. Управление осуществляется на основе информации об относительном положении ближайших БА в соответствии с заданными законами взаимодействия и текущего выполняемого задания. Выбор тех или иных законов взаимодействия БА находится в руках проектировщика. В ходе выполнении работы был разработан программный модуль, позволяющий моделировать управление группой БА в виде трех агрегатных состояний. А также была исследована устойчивость группы как системы материальных точек при различных законах их взаимодействия.

## Список литературы

**[Дегтярев и др., 2007]** Дегтярев Г.Л., Маликов А.И., Сиразетдинов Т.К. Сиразетдинов Р.Т. Исследования по устойчивости и управлению в КГТУ им. А.Н. Туполева-КАИ. Вестник КГТУ им. А.Н. Туполева, 2007. №3, с.90-97.

**[Каляев и др., 2010]** Каляев И. А., Капустян С. Г., Гайдук А. Р. Самоорганизующиеся распределенные системы управления группами интеллектуальных роботов, построенные на основе сетевой модели / Управление большими системами. Специальный выпуск 30.1 "Сетевые модели в управлении". М.: ИПУ РАН, 2010, с.605-639.

**[Миронов, 2013]** Миронов А.Б. Управление большими группами беспилотных аппаратов на основе имитации агрегатного состояния вещества. Науч. рук. Сиразетдинов Р.Т. // «XXI Туполевские чтения (школа молодых ученых)»: Международная молодежная научная конференция, 19-21 ноября 2013 г.: материалы конференции. – Т. I. – Казань: Изд-во Казан. Гос. Техн. Ун-та, 2013. с. 341-343.

**[Поспелов, 2008]** Поспелов Д.А. Ситуационное управление: теория и практика. –

М.: Наука, 1986.

**[Сиразетдинов, 1994]** Сиразетдинов Р.Т. Математическое моделирование развития системы однотипных объектов с учетом интенсивности их эксплуатации (На примере самолетно-вертолетного парка). Изв. ВУЗов. Сер. "Авиационная техника". Казань, 1994, №1, с.63-68.

**[Сиразетдинов, 1998]** Сиразетдинов Р.Т. Математическое моделирование мощности инфраструктуры сложных систем. Известия академии наук. Теория и системы управления, 1998. №3, с.96-104.

**[Сиразетдинов, 1997]** Сиразетдинов Т.К. Оптимизация систем с распределенными параметрами. М.: Наука. 1977, 480 с.

**[Сиразетдинов, 1987]** Сиразетдинов Т.К. Устойчивость систем с распределенными параметрами. Новосибирск: Наука, 1987, 232 с.

**[Тарасов, 2002]** Тарасов В.Б. От многоагентных систем к интеллектуальным организациям. М.: Эдиториал УРСС, 2002, 352 с.

**[Bassler, 1999]**Bassler, B. L. (1999). How bacteria talk to each other: regulation of gene expression by quorum sensing. Current Opinion in Microbiology, 2(6):582–587.

**[Duarte, 2014]**Duarte, M., Oliveira, S. M., and Christensen, A. L. (2014b). Hybrid control for large swarm of aquatic drones. In 14th International Conference on the Synthesis & Simulation of Living Systems (ALIFE), pages 785–792. MIT Press, Cambridge, MA.

**[Duarte, 2011]** Duarte, M., Oliveira, S., and Christensen, A. L. (2011). Towards artificial evolution of complex behavior observed in insect colonies. In 15th Portuguese Conference on Artificial Intelligence (EPIA), pages 153–167. Springer, Berlin, Germany.

**[Rodrigues, 2008]**Rodrigues, T., Duarte, M., Oliveira, S. M., and Christensen, A. L. (2015). Beyond onboard sensors in robotic swarms: Local collective sensing through situated communication. In 7th International Conference on Agents and Artificial Intelligence (ICAART). SciTePress, Lisbon, Portugal. In press.

**[Schmickl, 2008]**Schmickl, T. and Crailsheim, K. (2008). Trophallaxis within a robotic swarm: bio-inspired communication among robots in a swarm. Autonomous Robots, 25(1-2):171–188.

**[Stamatis, 2008]**Stamatis, P. N., Zaharakis, I. D., and Kameas, A. D. (2009). A study of bio-inspired communication scheme in swarm robotics. In 7th International Conference on Practical Applications of Agents and Multi-Agent Systems (PAAMS), pages 383–391. Springer, Berlin, Germany.

**УДК 004.896; 681.786.2**

# ИНФОРМАЦИОННАЯ СИСТЕМА ИНТЕЛЛЕКТУАЛЬНОГО БЕСПИЛОТНОГО АВТОМОБИЛЯ “АВТОНИВА”

В.Е. Павловский (*vlpavl@mail.ru*),
К.И. Кий (*konst.i.kiy@gmail.com*),
И.А. Орлов (*orlovbel@gmail.com*),
А.П. Алисейчик (*atooxa@gmail.com*)
Институт Прикладной Математики им. М.В. Келдыша РАН,
Москва

**Аннотация**. В работе описаны функции и главные алгоритмы подсистемы СТЗ АвтоНИВА, работающей с цветными телевизионными изображениями. Данная подсистема основана на новом, специальном сжатом описании каждого кадра, которое позволяет решать задачи нахождения и распознавания объектов в реальном времени, минимизируя обращения к исходному массиву изображения на стандартном персональном компьютере. Описана система выделения препятствий на пути движения автомобиля по дальномерным данным. Работа выполнена в рамках проекта РФФИ.[1]

**Ключевые слова**: автономное вождение, сегментация изображений, понимание изображений.

## Введение

С середины 80-х годов прошлого века начался интенсивный поток работ по созданию систем компьютерного зрения для обеспечения беспилотного движения автомобилей по дороге. В конце 80-х несколько исследовательских групп (включая одну советскую группу) смогли реализовать автономное движение по дорогам. В последнее время имеются значительные достижения в создании роботов-автомобилей для движения по дорогам общего пользования. Опытные образцы автомобилей-роботов имеются у многих ведущих фирм производителей автотранспорта. В частности продвинутые образцы автомобилей-роботов имеются у Daimler, Google и других компаний. Google-vehicle регулярно

[1] Работа выполнена при финансовой поддержке РФФИ (проект № 16-08-00880).

ездит по дорогом, и отчеты о его поездках можно найти на сайте компании Google. Несмотря на большие достижения, имеются направления развития компьютерного зрения, которые требуют дальнейшего развития и являются высоко актуальными. Это связано как с расширением условий, в котором возможно использовать авто-водителя и увеличением списка задач решаемых системой, так и с обеспечением значительного удешевления использованного оборудования. В работе описана подсистема СТЗ беспилотного автомобиля “АвтоНИВА” для обеспечения автономного движения по дорогам, основанная на новом подходе к сегментации, описанию, и пониманию цветных изображений.

## 1 Конструкция информационной системы

Конструкция информационной системы приведена на рис.1.

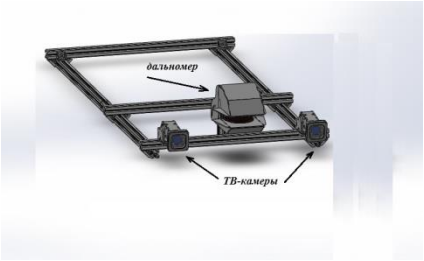



Рис.1. Конструкция информационной системы

Как показывает рис.1, система имеет специальную раму, на которой закреплены дальномер и две телекамеры. В качестве дальномера применен лазерный дальномер SICK, ТВ-камеры – профессиональные камеры VEC-255-IP с Ethernet-протоколом (2.0 Mpix). Рама крепится на крыше автомобиля, а сенсоры крепятся на раме так, что можно легко менять их позиции и настраивать систему на наиболее эффективную работу.

## 2 Задачи и принципы построения подсистемы СТЗ, основанной на обработке цветных изображений

Предлагаемые алгоритмы понимания дорожных сцен основаны на методе геометризованных гистограмм, описанном в [Kiy, 2010], [Kiy, 2015a], [Kiy, 2015b]. Данный метод является далеким развитием метода, примененного одним из автором данной статьи в успешном советском проекте по автономному движению по дорогам [Kiy et al., 1995].

## 2.1 Теоретические основы алгоритмов понимания дорожных сцен в СТЗ

С помощью техники, разработанной в [Kiy, 2010], [Kiy, 2015a], [Kiy, 2015b], каждому цветному изображению ставится в соответствие граф цветовых сгустков *STG*. Для построения *STG* изображение разбивается на полосы одинаковой ширины, параллельные горизонтальной или вертикальной оси плоскости изображения *Os*. Каждая полоса описывается множеством цветовых сгустков. Каждый цветовой сгусток $b$ характеризуется следующими параметрами: 1. интервал $[beg_b, end_b]$ – начало и конец $b$ на оси *Os*; $\Delta_H{}^b = [H_{min}{}^b, H_{max}{}^b]$ и $H_{mean}{}^b$ – диапазон и среднее значение цветового оттенка $b$; $\Delta_S{}^b = [S_{min}{}^b, S_{max}{}^b]$ и $S_{mean}{}^b$ – диапазон и среднее значение цветового насыщения; $\Delta_I{}^b = [I_{min}{}^b, I_{max}{}^b]$ и $I_{mean}{}^b$ – диапазон и среднее значения полутоновой компоненты, и мощность сгустка $Card^b$ (приблизительно, число точек в прообразе интервала в полосе, которые имеют цветовые характеристики, принадлежащие диапазонам цветового сгустка). Неформально, каждый сгусток дает описание некоторой части реального объекта в полосе, его проекцию на ось *Os* и описание значений численных характеристик этой части объекта. *STG* можно интерпретировать геометрически с помощью наложения отрезков его сгустков ($[beg_b, end_b]$) на центральную линию соответствующей полосы. Описание изображений с помощью цветовых сгустков сжимает информации об изображении с миллионов пикселов до нескольких сотен цветовых сгустков. Определяется подмножество доминирующих цветовых сгустков, отношение соседства между ними и определяется степень контрастности соседних цветовых сгустков. Многие задачи по поиску ориентиров и объектов в кадре, можно переформулировать строго как задачи поиска некоторых абстрактных объектов на графе цветовых сгустков. Для этих целей в [Kiy, 2015a] введены понятия левых и правых контрастных кривых на *STG* и определен двудольный граф левых и правых контрастных кривых *LRG*. Если изображение разбито на горизонтальные полосы, то неформально левая или правая контрастная кривая есть цепочка цветовых сгустков в соседних полосах с подобными цветовыми характеристиками. Отметим, что левые или правые концы цветовых сгустков меняются от полосы к полосе “непрерывно”, и соседние в той же полосе слева или справа цветовые сгустки имеют контрастные цветовые характеристики. Левая и правая контрастные кривые соединены ребром в *LRG*, если часть сгустков, их образующих, можно соединить цепочками других цветовых сгустков в одной и той же полосе без контрастных переходов. Связные компоненты в *LRG* определяют контура на изображении, которые являются границами областей и снабжены информацией о цветовых параметрах этих областей. Рисунок 2 (левая часть) показывает цветовые

сгустки, наложенные на средние линии полос. Набор левых и правых контрастных кривых из связной компоненты определяет форму полученной области.

Правые и левые контрастные кривые, соответствующие дороге и области растительности представлены на Рис. 2. Границы растительности на изображении, образованном цветовыми сгустками, выделены красными вертикальными отрезками. В данном случае они совпадают с границами дороги. Заметим, понятие левой (правой) контрастной кривой гораздо шире обычных контуров, так как они содержат информацию об области, лежащей слева (справа) от контура, определяемого границами отрезков цветовых сгустков.

На множестве доминирующих цветовых сгустков удается задать структуру полной упорядоченности (перенумеровать доминирующие цветовые сгустки от 0 до некоторого *k*). Построенное упорядочивание (сгустки перенумерованы с сохранением отношения соседства) позволяет построить граф соседства *ADG* (adjacency graph) для построенных контрастных объектов в *STG*. Граф *ADG* позволяет собирать сложные реальные объекты, которые состоят из разнородных частей. В этом графе устанавливаются отношения не только между объектами, которые непосредственно граничат друг с другом, но и между объектами изображения, разделенными заслонениями от других объектов. Например, с помощью графа *ADG* можно восстанавливать связи между частями дороги, связь между которыми прервана заслонениями от других транспортных средств, участвующих в движении. Новым результатам, связанным с детальным построением и применением графа *ADG* посвящена работа автора, находящаяся в печати [Kiy, 2017].

Рис. 2. Цветовые сгустки с выделенной границей растительности и левые и правые контрастные кривые, соответствующие дороге и растительности.

## 2.2. Объекты и ориентиры, выделяемые при анализе дорожных сцен и алгоритмы для их нахождения

На основе предложенной теории в СТЗ робота “АвтоНИВА” разрабатываются алгоритмы поиска ключевых объектов и ориентиров нескольких типов. К ключевым объектам относятся полотно дороги, разметка на ней, другие участники движения, характерные участки на них

(габариты, сигналы поворота, торможения), растительность в окрестности дороги, другие объекты в ее окрестности и область неба в кадре.

Также разрабатываются алгоритмы для поиска светофоров и понимания их состояния и алгоритмы поиска окрашенных конусов для обозначения строительных работ на дороге. Поскольку робот-автомобиль предназначен для движения по стандартным российским дорогам, включая проселочные дороги, это предъявляет особые требования к набору ключевых элементов, характеризующих дорогу, так как разметка может отсутствовать или быть сильно поврежденной ввиду климатических условий (особенно в конце зимы или весной). Также в виду присутствия других участников движения, значительные части дороги могут быть заслонены. Поэтому существенное внимание уделено анализу окрестности дороги.

Используя представление цветного изображения дорожной сцены с использованием графов *STG*, *LRG* и *ADG*, разработаны и реализованы в виде программного комплекса системы рассуждений для поиска области дороги, растительности в окрестности дороги, ее границы в окрестности дороги, строительных конусов на дороге.

В общем случае, каждый из объектов (растительность, дорога, небо) может состоять из нескольких разнородных контрастных объектов и не может быть выделен заданием каких-то порогов цветовых или яркостных характеристик. Для неба эти разнородные контрастные объекты состоят из кусков чистого неба, освещенных разным образом, зон облаков, имеющих разную плотность и также освещенных разным образом. Кроме того, яркость части облаков может быть ниже, чем яркость дороги смыкающейся с ней (при движении на подъем), и, как правило, она меньше окружения дороги из-за снега зимой. Особенно сложной становится задача выделения неба в городских условиях. Аналогичным образом, дорога с лужами, влажными пятнами, пятнами грязи, трещинами и дефектами покрытия также является сложным объектом для распознавания. Например, лужи, расположенные у края дороги, могут отражать растительность и иметь ее цветовые и яркостные характеристики. Влажные области в конце дороги также могут иметь цветовые и яркостные характеристики участков неба и фактически с ними смыкаться. Только тонкий анализ формы (сходящийся характер границ участка дороги) и анализ соседних объектов (тонкая полоска растительности между дорогой и небом) позволяют различить эти разные объекты. Сегментация, выделение контрастных объектов на изображении с помощью методов, представленных в [Kiy, 2010], [Kiy, 2015a], [Kiy, 2015b] позволяет найти достаточно большие относительно однородные части реальных объектов (дорога, растительность, лужи, части неба, и т.д.), описать их форму и цветовые и яркостные характеристики. С

помощью разработанной теории удается описать граф пространственных связей (отношение соседства) между построенными частями объектов. Построенные системы рассуждений взвешивают данные о положении объектов в кадре, их цветовых и яркостных характеристиках, отношении соседства, чтобы собрать цветовые сгустки, соответствующие области дороги, растительности или неба. Например, область, по цветовым параметрам близкая к небу расположенная в верхней части кадра, но имеющая сходящиеся границы и ”накрытая” сверху растительностью, будет отнесена к дороге. А область с расходящимися границами и лежащая над растительностью и имеющая сверху области неба будет отнесена к небу. Зона растительности, имеющая в достаточно близкой окрестности зону дороги и с противоположной стороны зону растительности такую, что границы обеих зон сходятся, считается зоной растительности, ограничивающей дорогу и т. д. Участки близкие к положению неба, имеющие близкие цветовые и яркостные характеристики, но имеющие прямолинейные границы и (или) структуры, похожие на окна, исключаются из области неба. Рисунок 2 показывает кадры областей неба в кадре, выделенных системой в разных видеопоследовательностях. Цветовые сгустки, принадлежащие к небу, нарисованы черным цветом.

Рис. 3. Цветовые сгустки, принадлежащие небу, наложенные на изображения.

## 2.3 Программная реализация и эксперименты с СТЗ

Пакет программ для СТЗ робота “АвтоНИВА” написан на C++ и работает в операционной системе Linux (Ubuntu 16). Для ввода изображений и записи результатов испытаний используются возможности OpenCV.

Разработанные программы отработаны на большом числе видеопоследовательностей, снятых при движении автомобилей. Программы визуализации результатов использованы при подготовки иллюстраций настоящей статьи.

# 3. Алгоритм обнаружения препятствий

При движении автомобиля по дороге, особенно по проселочной дороге, на пути могут возникать отдельные препятствия. Для их

обнаружения в комплексе "АвтоНИВА" применен алгоритм, разработанный для использования дальномерных данных. Этот алгоритм реализован как эвристический продукционный алгоритм, он может настраиваться на технические параметры мобильного аппарата, в частности, на параметры автомобиля "НИВА" (размеры и т.п.).

## 3.1. Описание алгоритма

Дальномер позволяет получить облако точек, до которых известны расстояния, а потому могут быть легко вычислены трехмерные координаты эти точек. Поскольку точки продуцируются дальномером в известном порядке, на них можно построить триангуляционную сетку *Ct*. Вычисления алгоритма производятся на этой сетке с целью обнаружения резких скачков направлений нормалей к треугольникам сетки *Ct* , характерных для препятствий.

Введем вектор $m_t$ маркировки треугольников сетки *Ct*. Этот вектор определяет раскраску сетки, при которой каждый треугольник с номером *k* может быть отнесен к одному из следующих типов: подстилающая поверхность ($m_t(k)=0$), недостоверный треугольник ($m_t(k)=-1$), препятствие ($m_t(k)=-2$), боковая (относительно расположения пары дальномер-препятствие) тень от препятствия ($m_t(k)=-3$), и тень от препятствия за препятствием ($m_t(k)=-4$). Опишем процедуру маркировки.

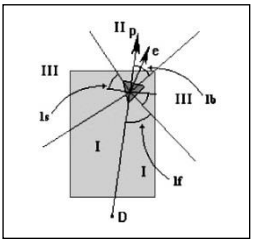



Рис.4. Классификационные зоны вокруг препятствия.

На рис.4 показаны проекция на плоскость *D'x'y'* (горизонтальная плоскость, проходящая через дальномер) некоторого тела (представляющего собой параллелепипед, в проекции - прямоугольник), проекции луча дальномера и нормали к выделенному треугольнику, а также передняя, задняя и две боковые классификационные зоны, обозначенные на рисунке цифрами I, II и III соответственно. Введенные зоны реализуют пороговую классификацию по направлениям нормалей к исследуемым треугольникам.

Треугольники сетки *Ct* классифицируются по нескольким порогам, по ним же классифицируются вершины этой сетки:

*c0* – порог угла наклона подстилающей поверхности,

*c1* – порог угла наклона нормали к вертикали,

*c2* – порог сонаправленности угла линии визирования с нормалью.

С помощью введенных пороговых величин будем далее классифицировать (раскрашивать) треугольники сетки *Ct* и их вершины. Для этого будем использовать два вектора признаков - треугольников $m_t$ и аналогичный для вершин, значения которых последовательно модифицируются.

Введем набор решающих правил *P = {P1, P2, P3, P4, P5}*, выполнение определенного сочетания которых позволит отнести текущий треугольник $T_k$ к одному из перечисленных выше типов. Правила *Pi* заключаются в выполнении следующих условий:

*P1* – нормаль сильно отклонена от вертикали (угол наклона нормали больше порога *c1*),

*P2* - проекция нормали на плоскость *D'x'y'* сонаправлена с проекцией луча обзора на ту же плоскость - нормаль направлена в классификационную зону III (рис.3),

*P3* - проекция нормали на плоскость *D'x'y'* противоположно направлена относительно проекции луча обзора на ту же плоскость - нормаль направлена в классификационную зону I,

*P4* - в треугольнике нет скачка по *z*,

*P5* - треугольник с предыдущим номером определен как препятствие.

Тогда окончательно вывод о наличии/отсутствии препятствий может быть сделан следующими продукционными правилами:

$$\textit{если} \quad P_2 \wedge \overline{P_4} \quad \textit{то} \quad m_t(k) = -4,$$

$$\textit{если} \quad \overline{(P_2 \wedge \overline{P_4})} \wedge (\overline{P_2 \vee P_3 \vee P_4}) \quad \textit{то} \quad m_t(k) = -3,$$

$$\textit{если} \quad \overline{(P_2 \wedge \overline{P_4})} \wedge \overline{(P_2 \vee P_3 \vee P_4)} \wedge (P1 \vee P3) \quad \textit{то} \quad m_t(k) = -2.$$

Поясним эти условия.

Если для треугольника $T_k$ одновременно истинно правило $P_2$ и ложно правило $P_4$, то такой треугольник маркируется (окрашивается) как тень за препятствием. Введенное условие означает, что нормаль к треугольнику направлена в классификационную зону II и в треугольнике имеется скачок высоты между его вершинами.

Если для треугольника $T_k$ не является истинным логическое произведение $P_2$ и $\overline{P_4}$, и если не является истинной логическая сумма правил $P_2$, $P_3$, $P_4$ то такой треугольник маркируется как боковая тень. Это означает, что треугольник будет окрашен как боковая тень, если не выполнено предыдущее условие, и одновременно при этом нормаль к

треугольнику не направлена в классификационные зоны I и II (т.е. направлена в зону III) и в треугольнике нет скачка высоты между его вершинами.

Если для треугольника $T_k$ не является истинным предыдущее условие, и если истинны условия $P_1$ или $P_5$, то такой треугольник маркируется как препятствие. Это означает, что треугольник будет окрашен как препятствие, если одновременно не выполнено предыдущее составное условие, и либо нормаль сильно отклонена от вертикали (в зону I), либо предыдущий треугольник уже окрашен как препятствие.

## 3.2 Эксперименты по моделированию

На следующем рисунке представлена карта, на которой нанесены 4 препятствия. На карте представлены результаты обработки измерений одного скана в проекции на плоскость $D'x'y'$ (ось $x'$ направлена горизонтально, а ось $y'$ направлена вертикально в плоскости рисунка). Точка $D'$ на рисунке соответствует проекции точки подвеса дальномера и является началом полярной сетки, определяющей сканирование.

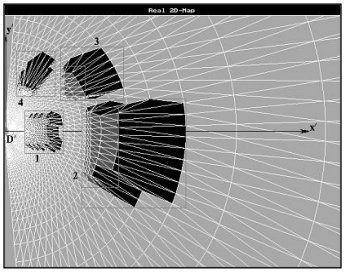



Рис.5. Карта препятствий.

При этом номерами 1 и 3 обозначены проекции измерений двух параллелепипедов (и их теней), определенных системой как препятствия. Объект с номером 2 представляет собой проекцию измерений призмы с основаниями, параллельными плоскости $D'x'y'$, и с наклоненной стороной, параллельной оси $y'$. Заметим, что угол наклона этой стороны больше значения предельного угла наклона поверхности, являющейся проходимой для робота (так были заданы соответствующие значения параметра *c0* и координаты вершин этой призмы), и поэтому данная сторона была определена как препятствие. Объект с номером 4 - это проекция измерений полусферы, центр которой лежит на плоскости сцены.

В ходе исследований различных примеров проведены эксперименты по использованию описанного алгоритма для зашумленных данных, приближенных к данным реального устройства. Эти эксперименты

показали устойчивую работу алгоритма в пределах до величин ошибок порядка 1-1.5% от величины исходной дальности. В указанном диапазоне алгоритм устойчиво выделял препятствия, причем их размеры и границы мало отличались от случая идеальных точных данных. При больших значениях ошибок алгоритм также выделял реальные препятствия, но при этом появлялись дополнительные "фантомные" препятствия - зоны, которые обозначались алгоритмом как препятствия, хотя фактически являлись проходимыми участками.

## Заключение

Эксперименты со зрительной системой показали ее полную работоспособность в различных реальных случаях. В настоящее время система отрабатывается на "АвтоНИВЕ".

Проведенные эксперименты по моделированию дальномерного алгоритма также подтвердили его работоспособность. Показана возможность применения алгоритма в указанном выше 1.5% диапазоне точности измерений. Представленные алгоритмы включены в комплекс "АвтоНИВА".

## Список литературы

**[Kiy, 2010]** Kiy K.I. A new real-time method for description and generalized segmentation of color images. // Pattern Recognition and Image Analysis. 2010. № 2. P.

**[Kiy, 2015a]** Kiy K.I. Segmentation and detection of contrast objects and their application in robot navigation. //Pattern Recognition and Image Analysis. 2015. № 2. P. 338-346.

**[Kiy, 2015b]** Kiy K.I. A new real-time method of contextual image description and its application in robot navigation and intelligent control, in *Computer Vision in Control Systems-2. Innovations in Practice*, Springer, Intelligence Systems Reference Library. Vol. 75, Ch. 5. P. 109-133.

**[Kiy et al., 1995]** Kiy K.I. A.V. Klimontovich, G.A. Buyvolov, Vision-based system for road following in real time. //Proc. International Conference on Advanced Robotics ICAR'95, Spain. Vol.1. P. 115–124, 1995.

**[Kiy 2017]** Kiy K.I. Geometrized histogram method of global real-time analysis of regions: application to recognition of road scenes and navigation of autonomous vehicles in *Computer Vision in Control Systems-3. Innovations in Practice*, Springer Book Series 8578, 2017, in press.

**УДК 004.896**

# КЛАССИФИКАЦИЯ ДОРОЖНЫХ СИТУАЦИЙ С ПОМОЩЬЮ БЕСПИЛОТНОГО ЛЕТАТЕЛЬНОГО АППАРАТА

Н.В. Ким (*nkim2011@list.ru*),
П.Д. Прохоров (*prokhorov_pd@mail.ru*),
Н.Е. Бодунков (*boduncov63@yandex.ru*)
Московский авиационный институт (национальный исследовательский университет) (МАИ)

**Аннотация.** В статье рассматривается использование беспилотных летательных аппаратов для автоматического мониторинга дорожного движения. Предложена методика классификации дорожных ситуаций по изображениям, полученных после дорожно-транспортного происшествия. Разработана иерархическая структура описания дорожной ситуации, наблюдаемой после произошедшего дорожно-транспортного происшествия. Для принятия решений о классе ситуаций предложено использовать продукционную модель представления знаний и соответствующую базу знаний. Приведен пример классификации ситуации по реальному изображению дорожного происшествия;

**Ключевые слова:** мониторинг дорожного движения, беспилотный летательный аппарат, транспортное происшествие, классификация ситуаций.

## Введение

В последнее время широкое внимание уделяется разработке алгоритмов и программ для обнаружения и оценки параметров движения транспортных средств с помощью беспилотных летательных аппаратов (БЛА) [Ashraf, 2011], [Kim, 2015a]. Использование БЛА для автоматического мониторинга дорожного движения может обеспечить существенное повышение пропускной способности дорожного движения, оптимизацию мероприятий по устранению последствий дорожно-транспортных (ДТП) происшествий, сокращению потерь, сопутствующих происшествиям и пр.

Целью статьи является разработка методики классификации (распознавания) дорожных ситуаций на основе анализа наблюдаемой сцены, полученной на борту БЛА после случившегося события.

Для оценки и прогноза развития возникших дорожных ситуаций требуется классификации этих ситуаций [Горелик, 2004]. В частности, для определения необходимых мероприятия при ликвидации их последствий.

В работе рассматривается способ классификации на основе выделения и анализа признаков (атрибутов) ситуации. Под признаками понимается наличие объектов интереса (например, транспортные средства и пешеходы) на сцене, их характеристики, а также отношения между ними, существенные для данной ситуации [Поспелов, 1986].

Следовательно, важным вопросом классификации является формирование описаний ситуаций, в которых должна содержаться информация, необходимая для принятия обоснованных решений. Предлагаемая методика основана на использовании онтологий [Liang, 2012], [Oberle, 2009] при описании ситуаций, а также применении каузальных отношений для вывода решений при классификации ситуаций.

## 1 Алгоритм классификации дорожных ситуаций

В рамках решаемых задач – обеспечения требуемой пропускной способности контролируемого участка дороги, нас интересует классификация, связанная с ликвидацией последствий возникшей ситуации.

Будем считать, что в процессе ликвидации последствий необходимо обеспечить:

оказание своевременной медицинской помощи пострадавшим,

минимизацию времени восстановления дорожного движения в штатном режиме,

снижения возможных потерь, связанных с вызовом средств помощи, которые не соответствуют возникшей ситуации.

Рассмотрим общий алгоритм классификации ситуаций.

На вход алгоритма подается принятое бортовой системой наблюдения (видеокамерой) сцена в виде отдельного изображения или видеопоследовательности.

Выходом алгоритма является решение о предполагаемом классе наблюдаемой ситуации, принятое на основе ее анализа.

Под анализом ситуации в работе понимается выделение объектов интереса, определение их параметров и межобъектных отношений, существенных с точки зрения классификации.

Ниже показана структура алгоритма классификации ситуаций, реализующей принципы анализа ситуации.

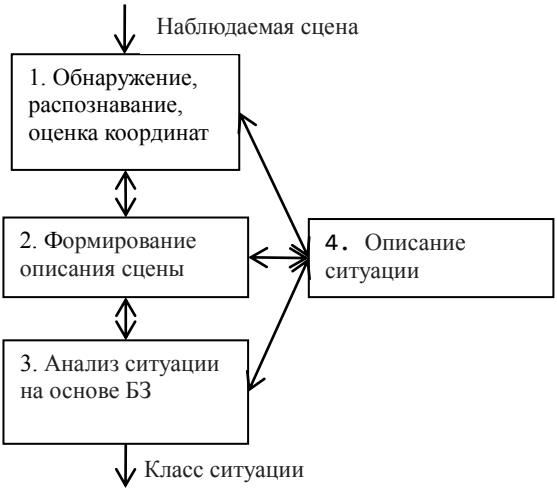



Рис. 1. Структура алгоритма классификации ситуаций

В блок 1 поступает видеоинформация о наблюдаемой сцене. На основе обработки и анализа принятых изображений производится обнаружение, распознавание, оценка положений и параметров объектов интереса. Перечень объектов интереса должен быть определен в блоке 4 (на основе заложенной в него онтологии). Существует множество алгоритмов, позволяющих решить эти задачи [Lienhart, 2002], [Yuping, 2011].

Например, для решения задачи обнаружения пешеходов на видеопоследовательности могут использоваться различные способы: метод Виолы–Джонса (Viola – Jones), корреляционный фильтр (ASEF), метод гистограмм направлений градиентов (HOG) и прочие.

На рисунке 2 приведен пример распознавания пешеходов на кадре видеопоследовательности с помощью HOG-дескрипторов [Dalal, 2005], основанных основанные на подсчете количества направлений градиента в локальных областях изображения.

Рис. 2. (кол-во ложных целей – 2, кол-во пропусков цели – 4)

Метод поддерживает инвариантность геометрических и фотометрических преобразований, за исключением ориентации объекта. Конечным шагом в распознавании объектов с использованием HOG является классификация дескрипторов при помощи системы обучения с учителем, к примеру, метод опорных векторов (SVM, Support Vector Machine).

В блоке 2 на основании полученных данных производится формирование описаний наблюдаемой сцены. При этом общая структура описаний определяется в блоке 4. В результате работы блока формируется описание, содержащее набор фактов, необходимых для классификации ситуации.

В блоке 3 производится анализ ситуации, результатом которого является определение искомого класса дорожной ситуации. Классификацию предлагается проводить на основе каузальной логики [Yuping, 2011], связывающей между собой причины и следствия исследуемых событий, с помощью заранее подготовленной базы знаний (БЗ). Для принятия решения используются факты, полученные из описания, сформированного в блоке 2.

В блоке 4 содержится общее описание ситуаций, определяющее структуру описаний объектов интереса, их свойств (атрибутов) и межобъектных отношений.

В общем случае возникающие чрезвычайные, особые ситуации могут классифицироваться по различным признакам. Например, по

происхождению или скорости развития.

В качестве основного принципа классификации выберем тяжесть последствий возникших дорожных ситуаций также, как в [Ashraf, 2011].

В рамках данного принципа в дорожные ситуации разделятся на 5 классов ($M = 5$).

Класс $x_1$ соответствует штатной дорожной ситуации.

Возникновение классов особых ситуаций $x_2, x_3, x_4, x_5$ связано со столкновениями транспортных средств (ДТП):

класс $x_2$ – особая ситуация, не приводящая к прямым материальным потерям, класс $x_3$ – особая ситуация, связанная с наличием незначительных материальных потерь. В этих ситуациях вызов специальных служб помощи (для ликвидации последствий происшествий) не требуется;

класс $x_4$ относится к аварийным ситуациям, сопровождаемым существенными материальными потерями, в частности, повреждениями транспортных средств высокой степени тяжести, но без человеческих жертв. Ситуация требует вызова технической помощи;

ситуация класса $x_5$ является катастрофической и сопровождается человеческими жертвами. В этой ситуации требуется вызов средств аварийно-спасательной службы, обеспечивающую оказание медицинской помощи.

## 2 Формирование описаний наблюдаемой сцены

На данном этапе необходимо разработать структуру описаний различных возможных ситуаций. Формируемое семантическое описание предназначено для удобства его дальнейшего анализа. При этом получаемые описания должны содержать полезную информацию, позволяющую в дальнейшем классифицировать эти ситуации.

Примем, что формируемые описания должны соответствовать ситуациям, возникающим после ДТП.

Сложность формирования подобных описаний состоит в том, что заранее неизвестно какие описания являются информативными для принятой классификации.

Поэтому на этом этапе структура описаний выбирается на основе экспертных оценок.

Описание носит иерархический характер и в зависимости от уровня иерархии делится на классы, подклассы и разделы различного уровня.

В общем случае могут использоваться различные виды описаний ситуации: пространственные, пространственно-временные, временные, каузальные.

Если предполагаемое решение поставленных задач основано на

анализе расположения объектов интереса, т.е. пространственных отношений между объектами, то целесообразно использовать пространственные описания. В рассматриваемой задаче классификации дорожных ситуаций (по их последствиям) данный вид описаний является наиболее важным.

В дальнейшем будет использоваться двумерное представление описания сцены с учетом направления и расстояния между объектами интереса.

В случае исследования процессов, связанных решением задач слежения за подвижными объектами, как в [Türmer, 2011], используется пространственно-временное описание ситуации.

Временные описания для рассматриваемой задачи являются не актуальными и в данном исследовании не рассматриваются.

Каузальные описания используются при анализе ситуаций (блок 3, рис. 1).

Описания состоят из следующих классов:

***«Наблюдаемая сцена»*** *(Observed Scene)* - общее описание наблюдаемой сцены (время, район наблюдения, условия наблюдения).

***«Объекты ДТП»*** *(Objects of TA)* - перечень объектов сцены (автомобилей, людей, препятствий и пр.) и их описания. В свою очередь описания также являются структурированными и содержат подклассы «Люди», «Автомобили» и т.п.

***«Внешние условия»*** *(External Conditions)* - описания внешних условиях, которые могут оказывать влияние на возникновение и развитие ДТП (погода, осадки, состояние дороги, особенности дороги).


Рисунок 3. Наблюдаемая сцена дорожно-транспортного происшествия

В соответствии с приведенной структурой сформируем описание ситуации на рисунке 3. На рисунке отмечены некоторые объекты участвующие в описании: трамвай («*V(1)*»), автомобили («*V(2)*», «*V(3)*»,

«*V(4)*»), группы людей («*GP(1)*», «*GP(2)*») и некоторые их свойства (тип объекта и состояние). Ниже представлен фрагмент описания наблюдаемой сцены в соответствии с предлагаемой структурой:

“Observed Scene”:
(FN(5777);TD(16-00);RN(3444);FD(7));
“Objects of TA”:
General Description: State: (SP(8)^SV(6)^R(1));
People: P(1): (State: PC(x:…; y:…)… ; SP(0)^B(N)^… );
P(2): (State: PC(x:…; y:…)… ; SP(0)^ B(N)^… );…
Relations: GP(P(1) ^ P(2)): (B(N) ^St());
GP(P(3) ^ P(4) ^ P(5)):(B(N)^St());…
Vehicles: V(1): (State: PC(x:…; y:…)… ;SV(7)^ B(N)^D(R:1) );
V(2): (State: PC(x:…; y:…)… ; SV(1)^ B(Un) ^D(R:7) ); …
Relations: GV(V(1)^V(2)): (D(1)^L(0));..
“External Conditions”:
Weather: (Wm); Road Conditions: ((Dr); Visibility(3); …);
где

*FN(...)* – номер кадра (файла), *TD(...)* – время, *RN(...)* – номер наблюдаемого участка дороги; *FD(...)* – направление на север (от центра изображения); *SP(24), SV(7), R(1)* – количество объектов определенного класса(люди, транспортные средства, дорога соответственно); *St, W* – состояние: «стоят», «идут», соответственно; *B(N)* – состояние (*N* – нормальное, *Un* – ненормальное, *Vr* –не определено);

*GP(P(1)^P(2)), GV(V(1)^V(2))* – группы объектов, в скобках объекты относящиеся к группе (один и тот же объект может относится к нескольким группам); *D(7), L(1)* – направление на объект (в восьмисвязной системе), расстояние до объекта*; PC(x:…; y:…), VC(x:…; y:…)* – координаты объектов;

*Weather: (Wm)* – погодные условия: «тепло»; *Dr* – дорожные условия: «Сухо»; *Visibility(5)* – видимость по 5-бальной шкале (0 –отсутствие видимости).

## 3 Анализ ситуаций

В блоке анализа ситуаций содержится база знаний (БЗ), в которой хранится информация об условиях возникновения особых дорожных ситуаций. На вход блока поступает описание текущей сцены, а на выходе – вывод о наиболее достоверном классе ситуаций.

Т.к. некоторые элементы описания представляются в нечеткой форме (например, состояние объектов) для построения БЗ используются нечеткие системы, основанные на нечетких правилах вида:

$$\boldsymbol{if}\ \mu_{A1} \cap \mu_{A2} \dots \mu_{AN}\ \boldsymbol{then}\ C = x_i,$$

где $A_1, \dots A_N$ – нечеткие утверждения (например, «поведение людей нормальное», «повреждения автомобилей не видны»), $\mu_{AN}$ – достоверность нечеткого утверждения, $C$ – вывод правила (например, вывод о возможном классе ситуации).

На основе имеющихся фактов, полученных из описания ситуации, определяются достоверности правил.

В таблице 1 представлены примеры правил из БЗ.

Таблица 1

| | A | B |
|---|---|---|
| | ТС(автомобиль) находятся неподвижно на проезжей части дороги и далее дорога свободна | ДТП, класс ситуации $x_2$ или $x_3$ или $x_4$ или $x_5$ |
| | ТС стоят вплотную на проезжей части дороги | класс ситуации $x_1$ или $x_2$, или $x_3$ или $x_4$ или $x_5$ |
| | Состояние ТС– нормальное (нет видимых повреждений) | класс ситуации $x_1$, или $x_2$, или $x_3$ |
| | Состояние ТС – не нормальное (есть видимые повреждения) | класс ситуации $x_3$ или $x_4$, или $x_5$ |
| | Два ТС стоят вплотную и перпендикулярно друг к другу | ДТП, класс ситуации $x_4$ или $x_5$ |
| | ДТП и одно из ТС повышенной опасности (трамвай) | класс ситуации $x_4$ или $x_5$ |
| | Достоверности $x_n$ и $x_m$ близки и m ≥ n | класс ситуации $x_m$ |

Для простоты примем порядок последовательного (по возрастанию номеров) использования условий, совпадающих с имеющимися фактами.

Для удобства поиска соответствующих условий *A(…)* в "*Description of Observed Scene"*они должны быть описаны по правилам описания наблюдаемой сцены. Например, пункт 5 из таблицы 2 будет соответствовать:

**If** (D(1)^L(0)) **then** $x_4$ , $x_5$;

При этом, для сцены на рисунке 3 достоверности утверждений *D(1)* и *L(0)* примут соответственно значения «1» и «0.9». Достоверность правила считается как минимальное значение достоверности, входящих в правило нечетких утверждений. Таким образом достоверность правила 5 будет «0.9».

Достоверности класса ситуации рассчитывается как сумма достоверностей правил, относящихся к этому классу:

$$p_{xi} = \sum_{l=1}^{L} \min(\mu_{A1}, \mu_{A2}, \dots \mu_{AN}).$$

где $p_{xi}$- достоверность *i-го* класса, $L$ – количество правил, относящихся к $i$-му классу.

На рассматриваемой сцене присутствует четыре ТС стоящих вплотную (достоверность правила 2 = 1). Среди них два ТС (*V(1)^V(2)*) расположены перпендикулярно друг к другу (достоверность правила 5 = 0.9). Одно из ТС участвующих в ДТП повышенной опасности – трамвай (достоверность правила 6 = 0.9). Таким образом, текущая ситуации с наибольшей достоверностью относится к классам $x_4$ (с существенными материальными потерями) или $x_5$ (с пострадавшими).

Достоверности полученных исходов одинаковы. В этом случае для принятия окончательного решения можно использовать правило 7, т.е. принимать наихудший вариант. Таким образом класс текущей ситуации - $x_5$.

## Заключение

Новизна представленной работы состоит в том, что здесь рассматривается случай, когда на борту UAV принимаются изображения, содержащие последствия произошедшего инцидента (ТА). Подобная классификация ситуаций часто затруднена отсутствием прямых признаков класса ситуации, например, видимых признаков повреждений транспортных средств или явных признаков, указывающих на наличие пострадавших людей.

Для описания ситуаций, содержащих максимально возможный объем полезной (для принятия решений) информации, разработана соответствующая структура описаний (“Description of Observed Scene”), а в качестве метода принятия решений предложено использовать продукционную систему.

## Список литературы

**[Ashraf, 2011]**Ashraf Qadir, William Semke, Jeremiah Neubert. “Implementation of an Onboard Visual Tracking System with Small Unmanned Aerial Vehicle (UAV).” International Journal of Innovative Technology & Creative Engineering (issn:2045-8711), Vol.1, No.10, October, 2011.

**[Dalal, 2005]** N. Dalal and B. Triggs, “Histogram of Oriented Gradients for Human Detection,” Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2005.

**[Feng, 2009]** Lin Feng, Lum Kai-Yew, Chen Ben M., Lee Tong H. “Development of a vision-based ground target detection and tracking system for a small unmanned helicopter.” Science in Chine Series F: Information Sciences, 2009, Springer.

**[Kim, 2014]** Kim N., Bodunkov N. «Computer Vision in Advanced Control Systems: Innovations in Practice», Volume 2, Editors M. Favorskaya, Lakhmi C. Jain, Springer 2014. – 295 p.

**[Kim, 2015a]** Kim N., Chervonenkis M. “Situational control unmanned aerial vehicles

for traffic monitoring.” Modern AppliedScience, Vol. 9, No. 5, May 2015, Special Issue//Canadian Center of Science and Education.ISSN (printed): 1913-1844. ISSN (electronic): 1913-1852

**[Kim, 2015b]** Kim N. Automated Decision Making in Road Traffic Monitoring by on-Board Unmanned Aerial Vehicle System. Indian Journal of Science and Technology, Vol 8(S10), December 2015

**[Liang, 2012**] Liang Li, Shuqiang Jiang, Qingming Huang. “Learning Hierarchical Semantic Description Via Mixed-Norm Regularization for Image Understanding.” Multimedia, IEEE Transactions ...> Volume:14, Issue:5, 2012, p. 1401 – 1413.

**[Lienhart, 2002]** Lienhart, Rainer, and Jochen Maydt. "An extended set of haar-like features for rapid object detection." Image Processing. 2002. Proceedings. 2002 International Conference on. Vol. 1. IEEE, 2002.

**[Oberle, 2009]** Oberle, D., Guarino, N., & Staab, S. (2009) What is an ontology?. In: "Handbook on Ontologies". Springer, 2nd edition, 2009.

**[Türmer, 2011]** Türmer, S.; Leitloff, J.; Reinartz, P.; Stilla, U. (2011): Evaluation of selected features for car detection in aerial images. ISPRS Hannover Workshop 2011, 14.-17. Jun. 2011, Hannover.

**[Yilmaz, 2006]** Yilmaz, A., Javed, O., and Shah, M. 2006. “Object tracking: A survey.” ACM Comput. Surv. 38, 4, Article 13. (Dec. 2006), 45 pages. http://doi.acm.org/10.1145/1177352.1177355

**[Yuping, 2011]** Yuping Lin, Qian Yu, Gérard Medioni. “Efficient detection and tracking of moving objects in geo-coordinates.” Machine Vision and Applications (2011), © Springer-Verlag 2010.

**[Zhang, 2012]** J. Zhang, L. Liu, B. Wang, X. Chen, Q. Wang, and T. Zheng, "High speed automatic power line detection and tracking for a UAV-based inspection," in International Conference on Industrial Control and Electronics Engineering (ICICEE), 2012, pp. 266-269

**[Горелик, 2004]**Горелик А.Л., Скрипкин В.А. Методы распознавания. М.: Высшая школа,2004.

**[Поспелов, 1986]** Поспелов Д.А. Ситуационое управление: теория и практика.- М.: Наука, - Гл.ред.физ.-мат.лит. 1986.-288с.

**УДК 629.7.05, 004.932.2, 519.687**

# БОРТОВОЙ УЗЕЛ ИСУ БЛА АВТОНОМНОГО ВЫПОЛНЕНИЯ ЗАДАЧИ ТОЧНОЙ ПОСАДКИ И СБРОСА ГРУЗА

А.Р. Гамаюнов (*agamayunov@le-talo.ru*)
Е.М. Притоцкий (*epritotskiy@le-talo.ru*)
М.С. Ходак (*khodakms@mail.ru*)
Владимирский государственный университет имени А.Г. и Н.Г. Столетовых, Владимир

**Аннотация**. В статье предлагается вариант реализации бортового узла, расширяющего базовые возможности типовой СУ БЛА. Рассмотрена задача локального позиционирования над визуальной меткой при посадке и сбросе груза. Описана техническая и программная реализация.

**Ключевые слова**: СУ БЛА, локальное позиционирование, точная посадка, сброс груза, техническая реализация.

## Введение

Полет любого БЛА разделяется на этапы: взлет, выполнение миссии и посадка. Управление полетом при выполнении любого из этих этапов - это сложный процесс, в ходе которого решается широкий круг задач, связанных с определением навигационных параметров полета и выдерживания необходимого пространственного положения. Эффективность действий при решении перечисленных задач определяется большим количеством условий, основными из которых есть своевременность, точность и связанные с ними вопросы построения траекторий движения, которые обеспечивают наиболее эффективное и безопасное достижение цели текущего этапа полета.

Этап посадки является наиболее ответственным участком полета, поэтому системы управления, реализующие автопосадку по заданной траектории патентуются крупными производителями БЛА, такими как DJI [1] и SKYCATCH [2], а также способы точной посадки разрабатывают военные, например, израильская фирма Sky Sapience с проектом HoverMast [3]. Точная посадка в последнем случае обеспечивается тем, что БЛА затягивается в бокс кабелем питания, при этом высота подъема

ограничивается длинной и весом кабеля.

В других случаях автопосадка позволяет приземлиться БЛА по заданным координатам, но точность обеспечивается системой навигации GPS и зависит от числа спутников, видимых над горизонтом. В идеальных условиях (чистом поле) типичная точность современных GPS-приёмников в горизонтальной плоскости при хорошей видимости спутников и использовании алгоритмов коррекции составляет от 1 до 8 м, но на практике точности для автономной, надёжной и безопасной посадки БЛА недостаточно. Для увеличения точности локального позиционирования малых автономных вертолётов применяют визуальные методы с погрешностью около 42 см [Merz, 2006], однако используя современные аппаратные и программные средства, результаты можно улучшить. Рассмотрим задачу увеличения точности позиционирования в приложении к поставленным практическим задачам.

## 1.1 Описание задачи

В последнее время всё чаще крупные организации стимулируют объединение отдельных исследователей в команды для решения практических задач данной области, организовывая соревнования и площадки для испытаний БТС. Одна из таких площадок – «Робокросс» предлагает участникам разработать беспилотный летательный аппарат, способный на открытой местности в автономном режиме взлетать, перемещаться по координатам, проводить сброс полезной нагрузки в цель на заданной координатами GPS площадке, вести объективный контроль попадания полезной нагрузки посредствам фото или видеофиксации и приземляться с погрешностью не более 1.5 м. Общая схема поставленной задачи показана на рисунке 1.

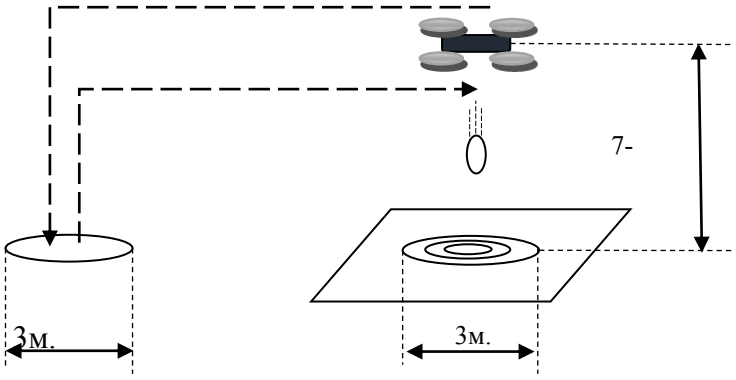



Рис. 1. Схема задачи.

Задачи автономного взлета, перемещения по координатам и видеофиксации с воздуха будем считать решенными [Bin, 2009]. Таким образом основные задачи, которые необходимо решить командам исследователей, это, во-первых, задача точной посадки, поскольку, как показала практика, точности позиционирования по данным GPS при посадке недостаточно, во-вторых, задача автоматического сброса полезной нагрузки на цель. Хотя заранее известны координаты сброса, их недостаточно для точного позиционирования. Для решения данной задачи необходимо обнаружить цель и точно над ней зависнуть.

# 1. Реализация

## 1.1 Описание платформы

Для решения поставленных задач будет использоваться мультироторный летательный аппарат вертикального взлёта. Задачу взлёта, полёта по точкам, управление механизмом сброса и приземления выполняет стандартный полётный контроллер. Необходимые датчики определения положения в пространстве обычно присутствуют в составе полётного контроллера. GPS и магнитометр выносится в отдельном корпусе как можно дальше от источников помех. Так же в отдельно снизу корпуса БЛА устанавливается акустический сонар.

Поскольку точности определения местоположения при посадке и сбросе полезной нагрузки, используя только данные GPS, недостаточно, то установленный на борту дополнительный компьютер определяет точное положение БЛА относительно визуальных меток посадочной площадки и площадки для сброса, используя данные с камеры и данные полётного контроллера [Ginkel, 2013].

Не менее важной задачей бортового компьютера является построение списка заданий для полётного контроллера во время исполнения автономной миссии. Для решения задачи видеофиксации на борту БЛА установлена вторая видеокамера, направленная в сторону земли.

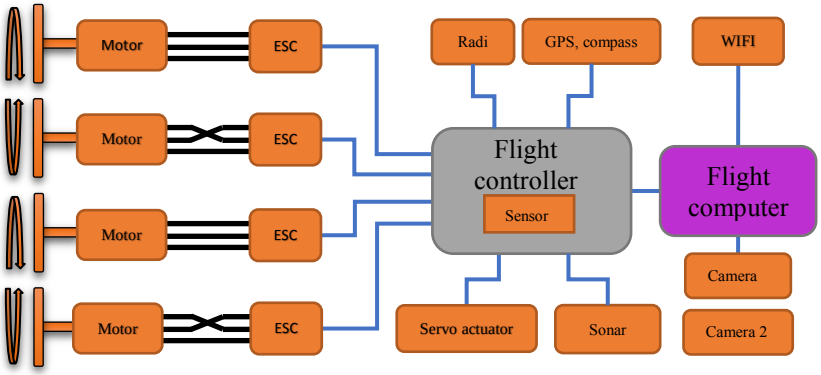



Рис. 2. Аппаратная часть.

На полётном контроллере установлена операционная система реального времени для обеспечения быстрой реакции на изменение положения БЛА в пространстве и корректировку его в соответствии с сигналом задания. Всю необходимую логику управления исполнительными механизмами и обработки данных датчиков реализует программа с открытым исходным кодом ArduCopter.

Бортовому компьютеру не обязательно выдерживать строгие временные интервалы, но решение его задач требует специализированных библиотек программного кода для распознавания образа посадочной площадки и площадки для сброса в видеопотоке с камеры, поэтому на нём установлена наиболее удобная для подобных разработок OS Ubuntu.

На бортовом компьютере в виде процесса демона работает наземная станция управления (GCS) в режиме прокси сервера между последовательным портом, к которому подключён полётный контроллер и управляющей программой. GCS организовывает логику общения с полётным контроллером согласно протоколу MAVLink. Таким образом управляющая программа получает удобный интерфейс управления БЛА. Данные с камеры управляющая программа получает по протоколу UVC. Для дальнейшей обработки графических данных управляющая программа использует библиотеку технического зрения OpenCV [Bradski, 2008].

Так же в системе имеется канал обмена данными по последовательному порту полётного контроллера для возможности ручного управления БЛА и доступ к полётному компьютеру по SSH через WIFI/Ethernet для отладки управляющей программы.

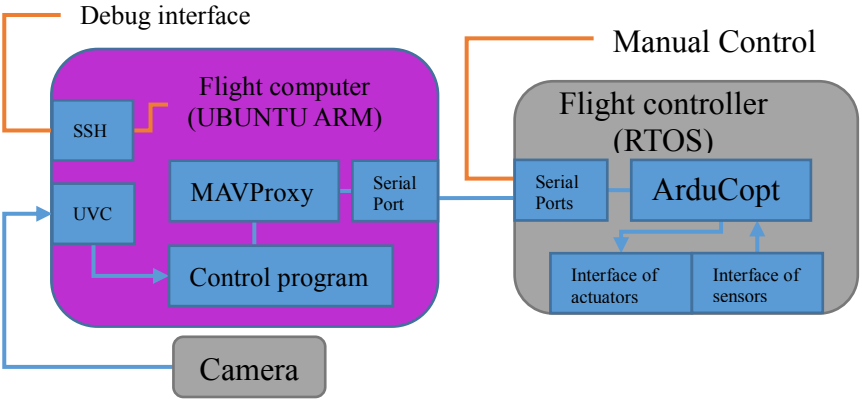



Рис. 3. Программная реализация.

## 1.2 Способ точной посадки

Предлагаемый способ точной посадки заключается в том, что бортовой компьютер БЛА при помощи алгоритмов компьютерного зрения обрабатывает временную последовательность кадров, получаемую с оптической камеры, установленной на БЛА, и содержащую данные об оптической метке, в точке посадки, для определения двух углов смещения. Бортовой компьютер также получает от полетного контроллера данные о двух углах наклона (крен и тангаж) и высоте, обрабатывает полученные данные и направляет сигналы управления на полетный контроллер для корректировки траектории и обеспечения точной посадки БЛА. Скорость регулирования и устойчивость системы зависит от параметров ПИД регулятора.

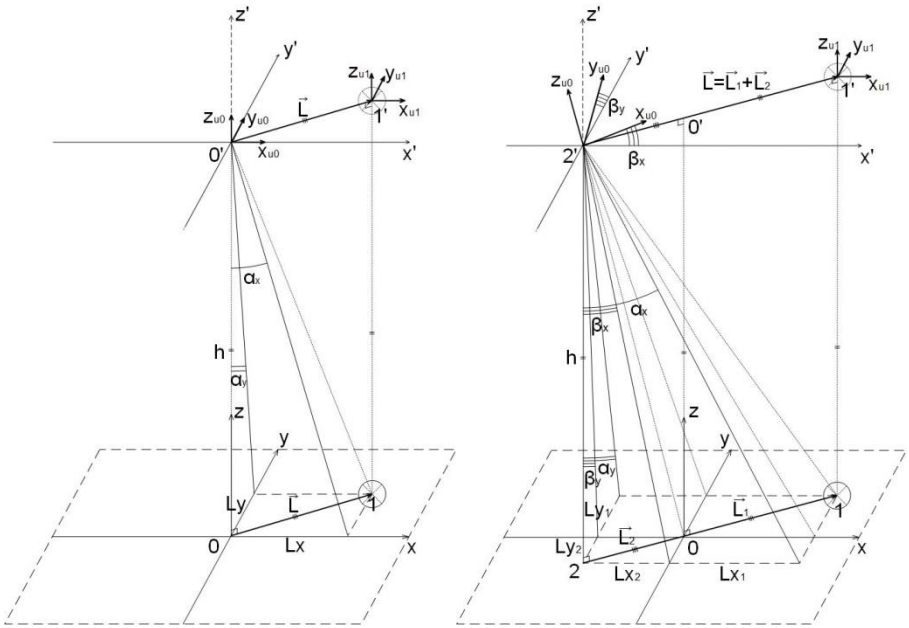



Рис. 4. Частный и общий случаи.

На рисунке 4 слева представлен частный случай, при котором углы крена и тангажа равны нулю. БЛА завис над оптической меткой на высоте h. Сигналы управления $L_x$ и $L_y$, направляемые от бортового компьютера на полетный контроллер, зависят только от углов смещения $\alpha_x$ и $\alpha_y$ и высоты h. Вектор смещения L задает направление движения в плоскости x'0'y'.

На рисунке 4 справа представлен общий случай, при котором углы наклона (крена и тангажа) $\beta_x$ и $\beta_y$ не равны нулю. БЛА движется путем изменения углов наклона над оптической меткой на высоте h. Сигналы управления зависят не только от углов смещения $\alpha_x$ и $\alpha_y$ и высоты h, но и от углов наклона $\beta_x$ и $\beta_y$. Вектор смещения L задает направление движения в плоскости x'2'y'.

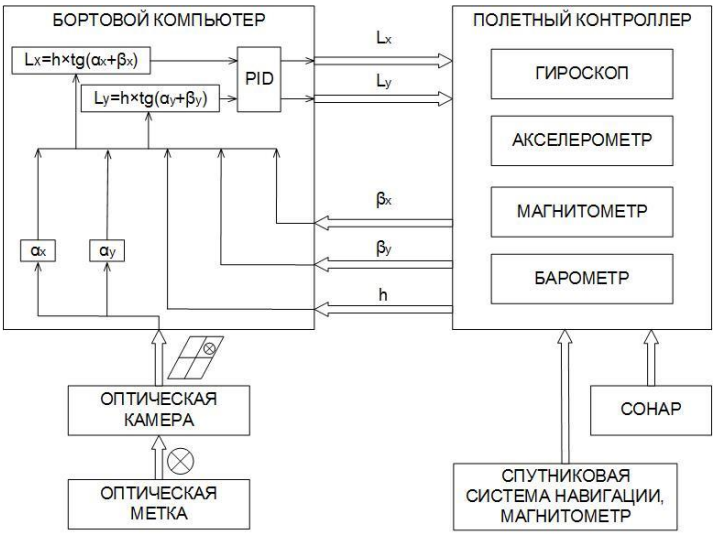



Рис. 5. Общая структурная схема взаимодействия отдельных элементов.

На рисунке 5 изображена схема взаимодействия отдельных элементов. Данные о высоте h, полученные со спутниковой системы навигации, барометра и сонара, и данные об углах наклона $\beta_x$ и $\beta_y$, рассчитанные исходя из данных с гироскопа, акселерометра и магнитометра, с целью увеличения достоверности полетный контроллер обрабатывает при помощи рекурсивного фильтра, например, фильтра Калмана. Данные об углах смещения $\alpha_x$ и $\alpha_y$ рассчитываются в результате обработки фильтрами библиотек компьютерного зрения OpenCV и операциями математической морфологии временной последовательности кадров. Бортовой компьютер формирует сигналы управления, на основании данных о векторе смещения, определенного в соответствии со следующими формулами:

$$L_x=h\times tg(\alpha_x+\beta_x) \quad (1)$$

$$L_y=h\times tg(\alpha_y+\beta_y) \quad (2)$$

где L – вектор смещения БЛА, $L_x$ – смещение БЛА по оси x, $L_y$ – смещение БЛА по оси y, h – высота, $\alpha_x$ – угол смещения по оси x, $\alpha_y$ – угол смещения по оси y, $\beta_x$ – угол наклона по оси x (крен), $\beta_y$ – угол наклона по оси y (тангаж).

Для быстродействия корректировки и достаточной точности посадки подбираются коэффициенты ПИД регулятора ручной настройкой, методом Зиглера–Никольса или методом CHR. Бортовой компьютер передает сформированные сигналы управления на полетный контроллер, который осуществляет корректировку курса БЛА во время его посадки в центр

оптической метки.

## 1.3 Способ точного сброса полезной нагрузки

Позиционирование над целью производится аналогично способу точной посадки, т.е. отслеживаются углы наклона и высота БЛА, а также отслеживается цель на изображении, получаемого с оптической камеры. При совмещении заданной цели с центром изображения в процессе позиционирования над ней и при углах наклона близких к нулю можно полагать, что БЛА находится прямо над целью и бортовой компьютер может подать команду на сброс груза.

Чем уже диапазоны разрешенных углов наклона и диаметр цели, тем больше возможность того, что они не выполнятся одновременно в заданный промежуток времени, но тем выше точность попадания груза в цель. Данные параметры следует подбирать экспериментально в ходе практических исследований.

При сбросе груза с ненулевой горизонтальной скоростью, задача превращается в классическую задачу бомбометания, как показано на рисунке 6. Траектория движения груза представляет собой параболу с вершиной в точке сбрасывания (3), а величина его относа без учёта сопротивления воздуха, которым можно пренебречь на малой высоте сброса, будет рассчитываться по формуле (4) [Шейгас, 2014]:

$$y = \frac{gx^2}{2V_0^2}, \tag{3}$$

$$\vec{A} = \overrightarrow{V_0} \times t, \tag{4}$$

$$t = \sqrt{\frac{2H}{g}} \tag{5}$$

где $t$ - время падения, $\overrightarrow{V_0}$ – вектор скорости БЛА в момент сброса груза.

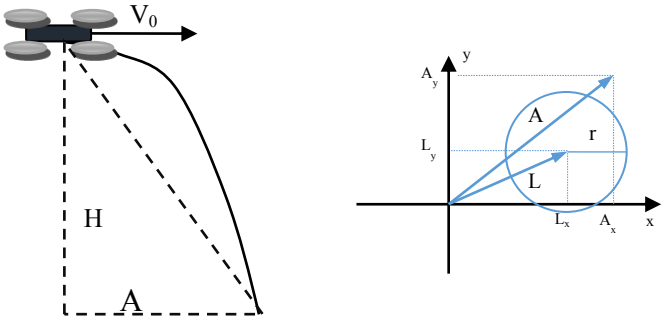



Рис. 6. Сброс полезной нагрузки.

Вектор L (1,2) указывает на центр оптической метки. r – это

доверительный радиус попадания. Как только конец вектора сноса A попадает в доверительный радиус попадания вокруг оптической метки сброса груза, полётный компьютер может давать команду на сброс полезной нагрузки.

## Заключение

Предложенный способ точной посадки и сброса полезной нагрузки являются практически применимыми, так как для их реализации подходят известные современные компоненты и открытые библиотеки программного кода. В ходе соревнований Робокросс 2015 авторами статьи был использован предложенный способ точной посадки, что позволило достигнуть погрешности при автономном приземлении около 15 см. Способ точного сброса полезной нагрузки будет испытываться в ходе соревнований Робокросс 2016.

## Список литературы

**[1]** US 20140236390 A1, МПК B64C29/00, B64D47/08, B64C19/00, Vertical takeoff and landing (vtol) small unmanned aerial system for monitoring oil and gas pipelines, Mohamadi, August 21, 2014.

**[2]** CN 204250382 U, МПК B64F1/02, Positioning mechanism and UAV (Unmanned Aerial Vehicle) base station using positioning mechanism, April 8, 2015.

**[3]** US20130233964 A1, МПК B64D17/80, H02G11/00, B64D25/00, G05D1/00, B64C37/02, Tethered aerial system for data gathering, Woodworth, Peverill, September 12, 2013.

**[Шейгас, 2014]** Факторы, влияющие на элементы траектории бомбы: учет в перспективных системах бомбометания // Збірник наукових праць Харківського університету Повітряних Сил, 2014. – №.3. – С.44-46.

**[Bin, 2009**] Bin H., Justice A. The design of an unmanned aerial vehicle based on the ArduPilot //Indian Journal of Science and Technology. – 2009. – Т. 2. – №. 4. – С. 12-15.

**[Bradski, 2008]** Bradski G., Kaehler A. Learning OpenCV: Computer vision with the OpenCV library. – " O'Reilly Media, Inc.", 2008.

**[Ginkel, 2013**] Robbert van Ginkel, Iris Meerman, Timo Mulder, Jorn Peters // Autonomous Landing of a Quadcopter on a Predefined Marker, 2013.

**[Merz, 2006]** Merz T., Duranti S., Conte G. Autonomous landing of an unmanned helicopter based on vision and inertial sensing // Experimental Robotics IX. – Springer Berlin Heidelberg, 2006. – С. 343-352.

**УДК 004.896**

# СИСТЕМА ПОЗИЦИОНИРОВАНИЯ МОБИЛЬНОГО РОБОТА ОТНОСИТЕЛЬНО РАЗМЕТКИ С ПРИМЕНЕНИЕМ СРЕДСТВ НЕЧЁТКОЙ ЛОГИКИ

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

**Аннотация**. Цель данной разработки — обеспечить позиционирование мобильного робота относительно края контрольной площадки при наличии строгих требований по ориентации и расстоянию до края по данным видеокамеры. Контрольная площадка по цвету резко контрастирует с окружающей местностью, что можно использовать для её распознавания. Позиционирование выполняется по данным видеокамеры с известными параметрами, установленной на роботе в известном положении, причём для упрощения формулировки задачи применены средства нечёткой логики.[1]

**Ключевые слова**: позиционирование, нечеткая система управления, PID-регулятор.

## Введение

Дистанционное зондирование Земли – это процесс постоянной спутниковой съемки планеты для сбора информации о различных процессах природного и антропогенного характера. Для достижения высококачественных результатов необходим постоянный контроль качества работы спутников. Помимо прочих методов, часто используется сравнение параметров участков поверхности (контрольных площадок), измеренных спутником, с их же значениями, но измеренными приборами на Земле, что позволяет корректировать результаты спутниковой съёмки, повышая достоверность собранных данных. Этот процесс осложняется тем, что измерительная аппаратура весьма дорого стоит, и оборудовать ею каждую из множества площадок невыгодно, поэтому до сих пор перенос и настройку этой аппаратуры выполняют люди-операторы. В связи с этим возникла задача разработки беспилотного транспортного средства, которое позволит автоматически переносить аппаратуру к нужной

[1] Работа выполнена при частичной поддержке гранта РФФИ 15-07-07483.

контрольной площадке и располагать её в соответствии с требованиями задачи.

## 1 Постановка задачи

Для функционирования обслуживающего беспилотного транспортного средства необходимо обеспечить его автоматическое перемещение в нужное положение относительно некоторой заданной заранее области (далее — контрольной площадки). Эта площадка выделяется на окружающей местности своей окраской, однородной с высокой степенью точности; её края высоко контрастны с грунтом как при естественном, так и при искусственном освещении. При этом характерные размеры площадки значительно больше характерных размеров робота, и вся доступная для движений робота местность считается горизонтальной. Нужное положение робота определяется не в абсолютных географических координатах, а по параметрам положения относительно края площадки ($d$,Θ), где $d$ — расстояние от центра симметрии робота до края площадки, Θ — угол между линией края площадки и передним направлением робота (рис. 1). При этом требуется обеспечить среднеквадратическое отклонение d менее 2 см, Θ – менее 1°.

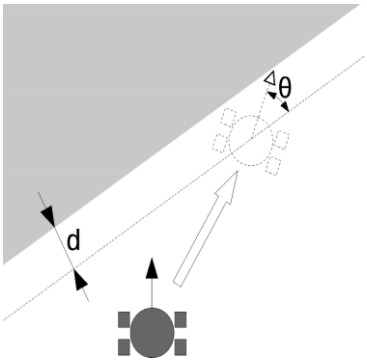



Рис. 1. Условия задачи.

Для позиционирования робота можно использовать различные подходы. Использование спутниковых систем навигации в данном случае осложнено тем, что полигоны с контрольными площадками часто располагаются в горной местности, где сигналы от части созвездия спутников недоступны либо искажены. В связи с этим возникла необходимость разработать автономную навигационную систему, частью которой и является реализуемая вспомогательная система позиционирования для беспилотного транспортного средства.

Известно много подходов к решению задач позиционирования (весьма полный обзор приводится, например, в [Borenstein, 1997]). Для задачи позиционирования робота по контрольной площадке оказалось целесообразным использовать изображение контрольной площадки, получаемое установленной на мобильный робот откалиброванной камерой. Для облегчения интеграции в существующую систему управления реализация готовой системы выполнена в виде компонента под управлением ROS.

Решение задачи можно разделить на этапы:

- Распознавание края контрольной площадки по данным видеокамеры;
- Определение положения края контрольной площадки относительно робота;
- Управление перемещением робота для получения точной ориентации относительно контрольной площадки.

Рассмотрим различные варианты решения данной задачи.

# 2 Алгоритм распознавания положения края контрольной площадки

Контрольная площадка имеет вид многоугольника, нарисованного на ровной горизонтальной поверхности краской, контрастной с этой поверхностью. Характерные размеры элементов контрольной площадки намного превосходят характерные размеры используемого робота, поэтому при позиционировании можно считать край прямой линией.

## 2.1 Обзор методов распознавания края контрольной площадки

Для определения границ контрольной площадки по изображению могут быть использованы методы одной из 2 групп – распознающие сплошные области (сегментация) либо их границы (распознавание края).

При сегментации изображение (целиком или полностью) разделяется на отдельные части, в данной задаче — контрольную площадку, грунт и, возможно, посторонние предметы. Из границ выделенных областей далее выбирается та, которая лучше всего подходит под некоторое заранее составленное описание искомой границы. Достоинство этих методов в том, что они позволяют с высокой надёжностью определить верную границу [Singh, 2014], так как сегментация через кластеризацию в рассматриваемых условиях должна, скорее всего, дать весьма качественные результаты (площадка и грунт примерно однородны, резко контрастируют друг с другом, переходных областей между ними нет). Однако, эти методы приводят к неоднократной обработке значительного

числа точек по всей площади изображения для формирования и обновления кластеров, что может плохо сказаться на скорости работы. Кроме того, при работе в реальных условиях могут появиться факторы, нарушающие однородность областей: отражение солнечного/лунного света, тени от облаков, загрязнения площадки и т.п.

Выделение границ, в данном случае — прямых линий, позволяет выделить участки изображения с перепадами яркости (либо другой характеристики) и затем сгруппировать их так, чтобы они составляли искомую фигуру. В поставленной задаче это означает, что после выделения всех прямых линий изображения остаётся только выбрать из них ту, которая наилучшим образом удовлетворяет условию задачи. Эти методы лучше подходят для поставленной задачи, так как требуют только различимой границы между областями, а не однородности самих областей [Jahne, 2005].

Известно множество методов выделения границ. Наиболее простые из них – алгоритмы, основанные на линейной фильтрации (методы Собеля, Прюитт, Шарра, Робертса), определяющие «толстую границу» без выделения векторных примитивов. Существуют методы дополнительной обработки, позволяющие выявить «тонкую границу» анализом малых окрестностей точек (детектор Кэнни и его вариации). Однако, в нашем случае (искомая прямая только одна, количество посторонних краевых точек мало) возможно не подавлять локальные экстремумы, а сразу решать задачу поиска.

Для выделения прямых часто применяют преобразование Хафа и основанные на нём техники. Однако, оно требует выделения памяти для хранения промежуточных результатов, причём с ростом числа искомых угловых коэффициентов (что может повысить точность) требования к памяти нарастают линейно. Предлагаемый метод лишён этого недостатка, так как требует память только для хранения граничных точек, что может быть желательно для реализации на встроенном компьютере целевого робота. К недостаткам предлагаемого метода относится непостоянное время работы на постоянных данных, однако, как будет видно из результатов моделирования, в условиях рассматриваемой задачи это не приводит к значительным задержкам.

## 2.2 Описание простейшего алгоритма распознавателя линий

Первый вариант алгоритма распознавания линий основан на выделении границ путём RANSAC. Последовательность действий при обработке следующая:

- отфильтровать ВЧ-шум;
- выделить границы одним из примитивных методов линейной фильтрации (пока использован фильтр Собеля);

- отделить точки границы от фона (например, пороговой фильтрацией с фиксированным порогом);
- для какой-нибудь пары разных точек границы:
  - рассчитать уравнение линии, проходящей через эту пару точек;
  - рассчитать евклидово 2–расстояние от каждой граничной точки до этой линии;
  - если вблизи этой линии оказалось достаточно много граничных точек, сделать вывод, что линия найдена; рассчитать её концевые точки;
  - иначе выбрать новую пару граничных точек;
- вернуть линию, если она найдена

Как видно, у этого алгоритма весьма много дополнительных параметров (фильтр ВЧ шума, фильтр выделения границы, метод расчёта близости точек к линии), однако, для простейшей реализации системы оценить данные параметры достаточно просто. Вопрос о калибровке камеры при этом следует решать отдельно.

Достоинством этого алгоритма является его производительность для данных условий. Число неудачных повторов обработки при наличии границы в кадре редко превосходит 1–2 (рис. 2).

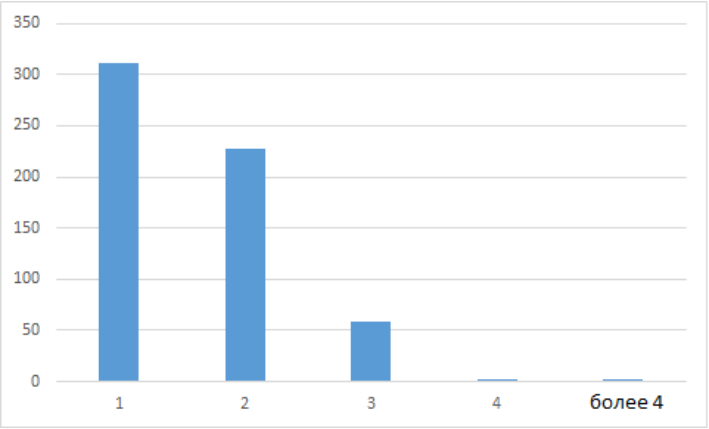



Рис. 2. Гистограмма числа повторов выбора граничной точки для успешного поиска линии на тестовой выборке.

Алгоритм был промоделирован с помощью Octave и проверен на наборе сгенерированных изображений (6 серий по 100 изображений с границей между чёрной и белой областью, гауссов шум по сериям: $\sigma$=0; 0,05; 0,1; 0,15; 0,2; 0,25 от амплитуды яркости) и полученных фотокамерой

в лабораторных условиях (рис. 3). Выяснилось, что данный метод работает достаточно надёжно при наличии резкой границы между тёмной и светлой областями (рис. 4, A); перепады текстуры без перепадов яркости не выявляются, что логично.

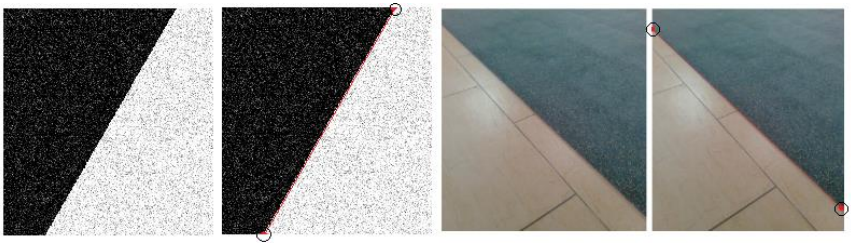


Рис. 3. Результаты распознавания линий на тестовом (слева) и полученном в лабораторных условиях (справа) изображениях.

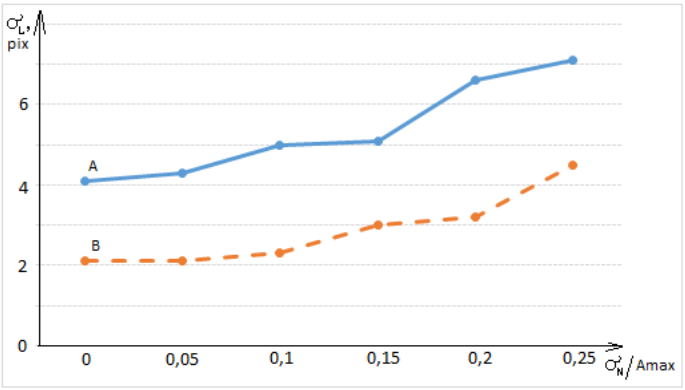



Рис. 4. Зависимость среднеквадратического отклонения положения конца найденной линии от шума на тестовом изображении (шум выражен через отношение среднеквадратического отклонения к амплитуде яркости) для исходного алгоритма (A) и дополненной версии (B)

Полученный метод оказался вполне работоспособным. Однако, в реальных условиях при управлении роботом проявилась нестабильность алгоритма RANSAC. Для повышения стабильности работы оказалось возможным дополнить алгоритм путём дополнительного поиска максимума градиента исходного изображения в малой локальной окрестности выбранных точек (размеры данной окрестности были выбраны на основе результатов моделирования начальной версии алгоритма, показанных на рис. 4). Эксперименты показали, что данный

алгоритм работает стабильнее исходного (рис. 4, В).

## 3 Определение положения края площадки относительно робота

В данной задаче по известным геометрическим параметрам мобильного робота с камерой требуется определить положение линии края площадки относительно робота. При этом также известны характеристики камеры (размеры изображения, а также фокусное расстояние либо обозреваемый угол) и наблюдаемое положение 2 точек, лежащих на краю площадки.

Задача в такой формулировке элементарно решается из геометрических соображений. Зададим локальную систему координат робота так, что ось X направлена по переднему направлению робота, ось Z — вверх, ось Y образует с ними правую тройку, то есть направлена влево; начало координат совпадает с центром симметрии робота. Пусть c – оптический центр камеры, d – направление её оптической оси, t и r - векторы, задающие плоскость кадра, имеющего размеры w и h (рис. 5).

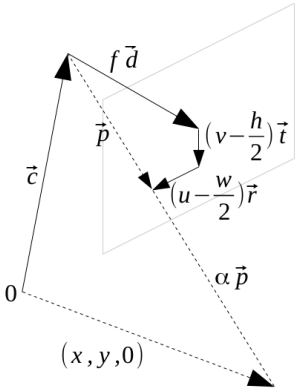



Рис. 5 Определение положения края площадки по его изображению

Направление **p** на точку кадра $(u,v)$ в координатах робота можно рассчитать как

$$\mathbf{p} = f\mathbf{d} + (v - 0.5h)\mathbf{t} + (u-0.5w)\mathbf{r}, \qquad (1)$$

из чего получаем её декартовы координаты:

$$x = \mathbf{c}_x - \mathbf{c}_z\,\mathbf{p}_x / \mathbf{p}_z\,,\ y = \mathbf{c}_y - \mathbf{c}_z\,\mathbf{p}_y / \mathbf{p}_z \qquad (2)$$

Зная декартовы координаты 2 точек линии, можно легко найти её положение относительно робота.

# 4 Управление движением робота

На основе изложенных ранее результатов распознавания был разработан алгоритм управления роботом. В его основе лежит PID-регулятор, позволяющий успешно привести робота в заданное положение; никакого запоминания местности роботом не проводится [Jazar, 2010]. Однако, так как область наблюдения камеры ограничена, то возможны положения камеры, при которых прямолинейное использование PID-регулятора приведёт к выходу границы из поля зрения. Например, если робот должен подойти к границе точно боком (т.е. $\Theta=0$), а камера смотрит в боковом направлении для точного позиционирования, то, когда робот увидит линию, он не должен двигаться к ней по кратчайшей траектории — ведь это приведёт к тому, что камера отвернётся от линии, и задача не будет решена. Поэтому оказалось целесообразным разделить пространство в окрестностях линии на две части — зону приближения и зону ориентации. Данные зоны было решено моделировать нечёткими множествами точек пространства в окрестностях границы. Графики функций принадлежности приведены на рис. 6.

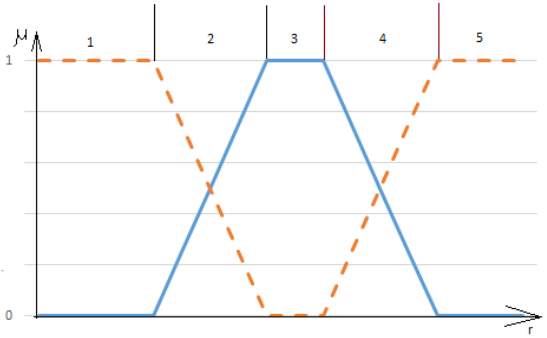



Рис. 6. Функции принадлежности точек пространства к зонам приближения (пунктир) и ориентации (сплошная линия); целевое расстояние находится в середине зоны ориентации.

Ширина зон приближения и ориентации определяется положением камеры на роботе таким образом, чтобы внутри зоны ориентации робот гарантированно не потерял из вида линию границы при условии, что он её увидел. Размеры переходных областей (2 и 4 на рис. 6) определяются заранее по скоростным характеристикам робота.

Нечёткое описание областей позволяет свести задачу к работе нечеткой системы управления [Seising, 2007].

Таким образом, алгоритм формирования управляющих сигналов можно описать следующим образом:

- сформировать управляющие сигналы системы, отвечающей за приближение робота к краю площадки.
- сформировать управляющие сигналы системы, отвечающей за правильную ориентацию робота.
- получить сумму сигналов первой и второй систем, взвешенных в соответствии с функциями принадлежности.

## 5 Реализация системы позиционирования

Система реализована в виде набора модулей ROS с применением средств библиотеки OpenCV. Моделирование тестового робота осуществлялось в среде Gazebo (внешний вид модели показан на рис. 7a, вид с камеры модели в её крайнем положении – 7b); в качестве реальной тестовой платформы использован DrRobot X80 с установленной видеокамерой.

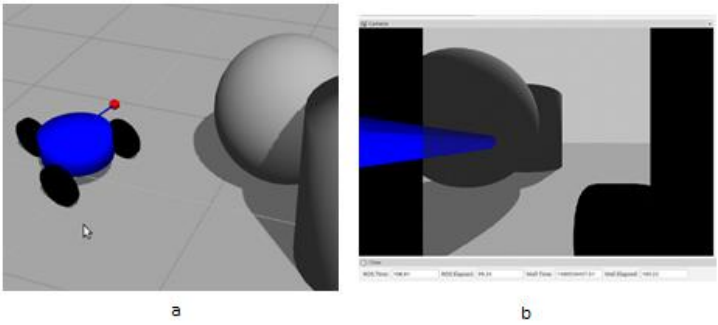



Рис. 7. Внешний вид модели (a) и вид с её камеры (b)

Тестирование алгоритма на модели показало, что при надлежащем подборе коэффициентов модель способна достичь требуемых показателей точности позиционирования за приемлемое время. Рис. 8 отражает некоторые зависимости результатов моделирования от его параметров.

Тестирование алгоритма выявления линий на реальном роботе показало, что данный алгоритм и в этом случае позволяет успешно решать поставленную задачу. В частности, обработка видеоданных с тестовой камеры (кадр 640x480, 30 кадров в секунду) проводилась в реальном времени; при этом система управления позволяла достичь заданной позиции за 2-4 секунды при начальной ошибке позиции 1.5-2м. Проблемы

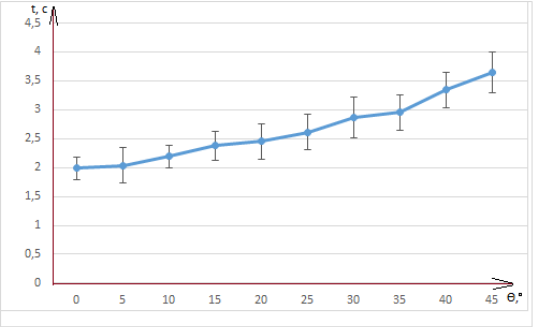



Рис. 8. Пример зависимости времени достижения цели от начальной ориентации робота при прочих равных условиях (для каждого угла выполнено 50 повторов моделирования)

с работой (устойчивые отказы распознавания линии) возникают при наличии на изображении длинных лишних границ (например, из-за посторонних предметов), в том числе не пересекающих искомую границу, а также при наличии бликов на контрольной площадке. Однако эти проблемы при соблюдении режима эксплуатации целевой системы окажутся незначительными (съёмка будет проводиться преимущественно в безлюдных местностях и в темное время суток).

## Заключение

В данной работе описаны разработка и реализация системы позиционирования мобильного робота с применением средств нечёткой логики. Данная система успешно решает поставленную задачу, обеспечивая позиционирование с требуемой точностью и скоростью.

## Список литературы

**[Borenstein, 1997]** J. Borenstein, H. R. Everett, L. Feng, D. Wehe. Mobile robot positioning — sensors and techniques // Invited paper for the Journal of Robotic Systems, Special Issue on Mobile Robots. Vol. 14 No. 4, pp. 231 – 249, 1997.

**[Jahne, 2005]** B. Jahne, Digital Image Processing. Springer Science & Business Media, 2005.

**[Jazar, 2010]** R.N. Jazar. Theory of Applied Robotics. 2nd edition. Springer, 2010. ISBN 978-1-4419-1749-2.

**[Seising, 2007]** Seising, R. The Fuzzification of Systems. The Genesis of Fuzzy Set Theory and Its Initial Applications . Springer-Verlag.

**[Singh, 2014]** I. Singh, D. Kumar. A Review on Different Image Segmentation Techniques // Indian Journal of Applied Research, v. 4, Issue 4, Apr 2014.

**УДК 681.513; 004.932**

# РАЗРАБОТКА СИСТЕМЫ ТЕХНИЧЕСКОГО ЗРЕНИЯ ДЛЯ СЕРВИСНОГО МОБИЛЬНОГО РОБОТА

Г.А. Прокопович (*rprakapovich@robotics.by*)
Объединённый институт проблем информатики
НАН Беларуси, Минск

**Аннотация**. Приводятся результаты экспериментов по разработке системы технического зрения, предназначенной для корректировки движения автономного мобильного робота по центру коридоров на основе данных от монокулярной камеры. Алгоритм выявления точки линейной перспективы основан на определении центра масс облака точек пересечения диагональных прямых, на которых лежат отрезки, найденные с помощью преобразования Хафа и образуемые краями и линиями цветовых переходов находятся в коридорах различных артефактов. Предложенный алгоритм протестирован на разработанном прототипе автономного мобильного робота с двухуровневой системой управления: нижний реализован на основе микроконтроллерной платы Arduino Mega, а верхний – на основе микрокомпьютера Raspberry Pi, управляющие программы для которых были смоделированы и сгенерированы в среде Simulink.[1]

**Ключевые слова**: автономный мобильный робот, система технического зрения, точка линейной перспективы, преобразование Хафа.

## Введение

По положительному примеру роботизации различных отраслей промышленности, одним из главных направлений автоматизации в логистике и социальной сфере является их частичная либо полная роботизация. На данный момент различными коллективами активно ведутся научно-практические работы по роботизации транспортно-складских задач, которые направленны на разработку алгоритмов пространственной ориентации и планирования оптимальных маршрутов автономными мобильными роботами (АМР), выполняющих транспортные и погрузочно-разгрузочные работы.

[1] Работа выполнена при финансовой поддержке БРФФИ (проект №Ф15УК/А-048).

Целью данной работы является разработка надёжных и простых способов распознавания двумерных образов в режиме реального времени, необходимых для управления движением АМР по середине коридоров. Для решения указанной задачи чаще всего используются сканирующие дальномеры в инфракрасном или ультразвуковом диапазонах, системы технического зрения (СТЗ) с одной или несколькими видеокамерами, а также другие сенсоры, позволяющие строить карты внутри помещений [Герасюто и др., 2014]. На основе данных измерения расстояний до близлежащих предметов, а также анализа изображений по характерным точкам приборы данного типа позволяют получить дву- или трёхмерные изображения окружающего пространства перед АМР. Однако они характеризуются большой стоимостью и отрицательным влиянием на здоровье персонала.

## 1 Программная часть разработанной СТЗ

На рис. 1 изображена функциональная схема разработанной СТЗ, предназначенная для управления движением АМР в стандартных комнатах и коридорах жилых и производственных помещений. Причём, при постановке технического задания разрабатываемой системы управления АМР ставились жёсткие условия: использовать в качестве сенсоров, необходимых для движения АМР вдоль коридоров, исключительно одну монокулярную RGB-видеокамеру. Полученные в блоке принятия решений управляющие сигналы подаются на нижний уровень системы управления АМР. Камера и блок принятия решений более подробно будут рассмотрены ниже. Пунктирной линией выделены этапы предварительной обработки.


Рис. 1. Функциональная схема предложенной СТЗ.

Предварительная обработка сенсорных данных является неотъемлемой частью современных систем управления, так как она позволяет в потоке избыточных данных найти те характерные элементы (реперы), на основе которых построены конкретные алгоритмы принятия решений. Наиболее важным этапом предварительной обработки сенсорных данных является этап сегментации, по результатам которого объём данных значительно сокращается, но зато увеличивается количество полезной информации. Далее рассмотрим этапы предварительной обработки более подробно.

## 1.1 Предложенная гипотеза для пространственного ориентирования АМР

Все артефакты, созданные человеком, отличаются исключительным свойством наличия прямых линий и углов, что очень редко встречается в живой природе. Поэтому, в качестве одного из способов движения АМР в жилых и производственных помещениях предлагается использовать точку линейной перспективы, состоящей из пересечения диагональных и горизонтальных линий. Так, если рассмотреть, например, коридоры больниц или складские помещения со стеллажами, то все они состоят из прямых плоскостей и горизонтальных линий: между полом и стенами, потолком и стенами, стеллажами и т.д.

Предложенный алгоритм [Прокопович и др., 2014] основан на оптическом явлении линейной перспективы – явлении мнимого искажения пропорций и формы тел при их визуальном наблюдении. В данной работе был использован линейный вид перспективы, рассчитанный на фиксированную точку зрения и предполагающий единую точку схода на линии горизонта, т.е. когда предметы уменьшаются пропорционально по мере удаления их от переднего плана. Поэтому на захваченном видеокамерой графическом изображении можно будет выделить параллельные линии вдоль оптической оси видеокамеры, которые пересекаются в общей точке, расположенной в идеале в центре изображения.

Следовательно, задача ориентирования АМР сводится к выделению на графических изображениях диагональных линий, образуемых краями и линиями цветовых переходов полов, стен, дверей, стеллажей и т.д., с последующим определением точки их линейной перспективы. Определив координаты точки перспективы конкретного помещения на цифровом изображении далее можно вычислять корректирующие коэффициенты для движения АМР. Причём, предложенный способ может использоваться для реализации задачи навигации роботов как внутри, так и вне помещений.

## 1.2 Предварительная обработка видеоданных

В данной работе представлены результаты экспериментов по разработке алгоритмов сегментирования сенсорных данных на основе преобразования Хафа [Гонсалес и др., 2006]. Преобразование Хафа предназначено для поиска в сенсорных данных различной природы объектов, принадлежащих определённому классу фигур с использованием процедуры голосования. Процедура голосования применяется к пространству параметров, из которого и получаются объекты определённого класса фигур по локальному максимуму в, так называемом, накопительном пространстве.

Преобразование Хафа – это преобразование координат всех ненулевых пикселей во входном бинарном изображении в абстрактное пространство параметров. Эти параметры точно задаются аналитическим уравнением необходимой формы (линии, окружности, эллипса и т.д.). В этом результирующем пространстве параметров обнаруживается задаваемое число пиков (глобальных максимумов), а затем соответствующие пространственные координаты преобразуются обратно в плоскость изображения в виде обнаруженного объекта (линии, окружности).

Наиболее распространенным является преобразование Хафа для обнаружения линий и кругов. Оно играет важную роль в решении задач навигации мобильной робототехники и понимании сцены. Например, двоичная версия границ изображения (контуры) [Форсайт и др., 2004] может служить в качестве входного значения для преобразования Хафа при обнаружении прямых линий. Произвольную линию перпендикуляра $\rho$ от начала координат и угла $\theta$ наклона нормальной линии от оси x можно представить в параметрической форме, заданной уравнением

$$\rho = x \cdot \cos(\theta) + y \cdot \sin(\theta).$$

В этом случае двумерное пространство координат изображения ($X$, $Y$) отображается в новом пространстве параметров линейного уравнения ($\rho$, $\theta$), также часто называемое Хаф-аккумулятором. $X$ и $Y$ координаты каждого ненулевого пикселя во входном бинарном изображении превращаются в одну пару параметров ($\rho$, $\theta$). В аккумуляторе соответствующая точка вычисленных координат $\rho$ и $\theta$ увеличивается на 1. Такая избирательная система является причиной того, что все пиксели, лежащие в одной прямой линии, создают пики в аккумуляторе с максимальным значением, соответствующим количеству пикселей. Неидеальная форма линии приводит к смазыванию пиков, но все равно она отличается от нижней окрестности.

## 2 Аппаратная часть разработанной СТЗ

В качестве прототипа АМР, оснащённого разработанной СТЗ и способного самостоятельно двигаться по центру комнат и коридоров, использовался полноприводный четырёхколёсный мобильный робот (рис. 2). Указанный АМР разработан на базе шасси Rover 5, оснащённого четырьмя инкрементными энкодерами и на борту которого были установлены монокулярная камера, раздельные платы нижнего и верхнего уровня, соединённые между собой по интерфейсу RS232, беспроводной канал связи с оператором на базе USB Wi-Fi модуля, дополнительный DC-DC понижающий преобразователь напряжения и 7.4В Li-Po батарея питания.

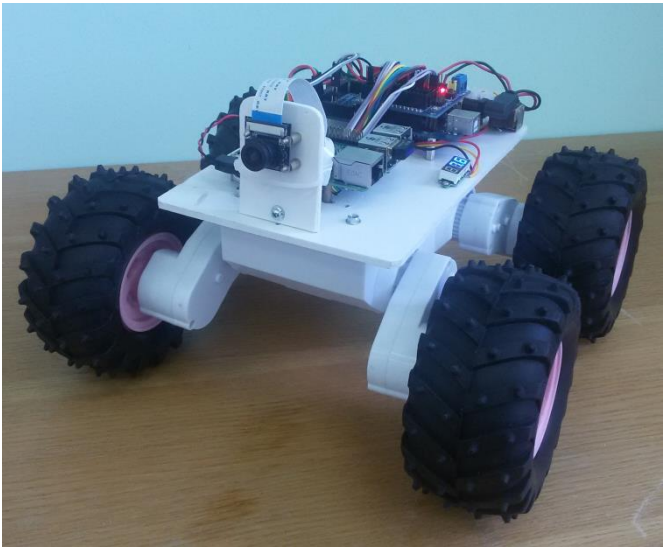


Рис. 2. Внешний вид разработанного АМР с двухуровневой системой управления.

## 2.1 Состав и структура нижнего уровня управления

Нижний уровень управления разработанной СТЗ, которая сама является основной частью системы управления АМР, представлен двумя сдвоенными мостовыми драйверами двигателей на основе микросхемы L298N и платой Arduino Mega, которая построена на микроконтроллере ATmega2560 и имеет 54 цифровых входа/выходов, 16 аналоговых входов, 4 последовательных порта UART, кварцевый генератор 16 МГц, USB коннектор и разъем питания. Основной функцией нижнего уровня является управление четырьмя двигателями постоянного тока, которое реализует:

- генерацию ШИМ-сигналов;
- реверсное управление;
- съём данных с инкрементных энкодеров;
- реализацию ПИД регулятора по скорости и углу поворота;
- контроль за электрическими показателями батареи питания.

## 2.2. Состав и структура и верхнего уровня управления

Верхний уровень управления представлен одноплатным микрокомпьютером Raspberry Pi, первоначально разрабатываемый для образовательных и научных целей. Одно из последних поколений Raspberry Pi 2 (далее RPi2) было выпущено в 2015 году. Второе поколение

основано на системе Broadcom BCM2836 на чипе с четырехъядерным процессором ARM Cortex-A7 и двухъядерным GPU VideoCore IV и с 1 Гб оперативной памяти. Эта небольшая, универсальная и высокопроизводительная платформа отлично подходит для прототипирования систем машинного зрения и решения задач мобильной робототехники.

В качестве монокулярной камеры использовалась плата расширения камеры для Raspberry Pi (RPiCam). RPiCam оснащена 5-Мп CMOS сенсором OmniVision 5647 с максимальным разрешением 2592 на 1944 пикселей и скоростью передачи 1080p / 30 кадров в секунду или 720p / 60 кадров в секунду.

Плата RPiCam имеет размер 25x20x9 мм и напрямую подключается к специализированному видео разъему на RPi2 платформе плоским многожильным кабелем. RPi2 содержит дополнительные интерфейсы HDMI, USB, RJ45, GPIO, аудио-разъем jack 3,5 мм и разъемы для подключения дополнительных устройств RPi.

# 3 Разработка управляющих программ в среде Simulink

Matlab является мощным инструментом для научных вычислений и моделирования технических систем [Герман-Галкин, 2008]. Он оснащен такими мощными библиотеками для решения задач компьютерного зрения и компьютерной обработки изображений, как Computer Vision Toolbox, Image Processing Toolbox и Image Acquisition Toolbox. Было решено реализовать указанные выше алгоритмы обработки изображений на RPi2, а управление двигателей на Arduino Mega с помощью среды Simulink. Инструменты среды Simulink позволяют проектировать приложения для компьютеров RPi и микроконтроллерных плат семейства Arduino с помощью дополнительных пакетов Simulink Support Packages для Raspberry Pi Hardware и Arduino Hardware.

## 3.1. Simulink-модель управления верхним уровнем

На рис. 3 изображены основные функциональные блоки Simulink-модели, которая реализует предложенный алгоритм нахождения точки перспективы в кадрах видеопотока на верхнем уровне управления:

- блок Preprocessing соответствует блоку «Фильтрация» (рис. 1*б*). Этот блок предназначен для преобразования цветового пространства изображения из RGB в градации серого с последующим выделением контуров объектов (рис. 4*б*), которое в данном случае выполнено с помощью стандартного преобразования с ядром Робертса;

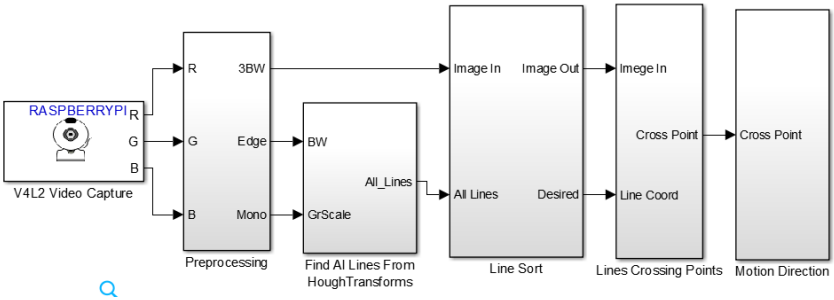
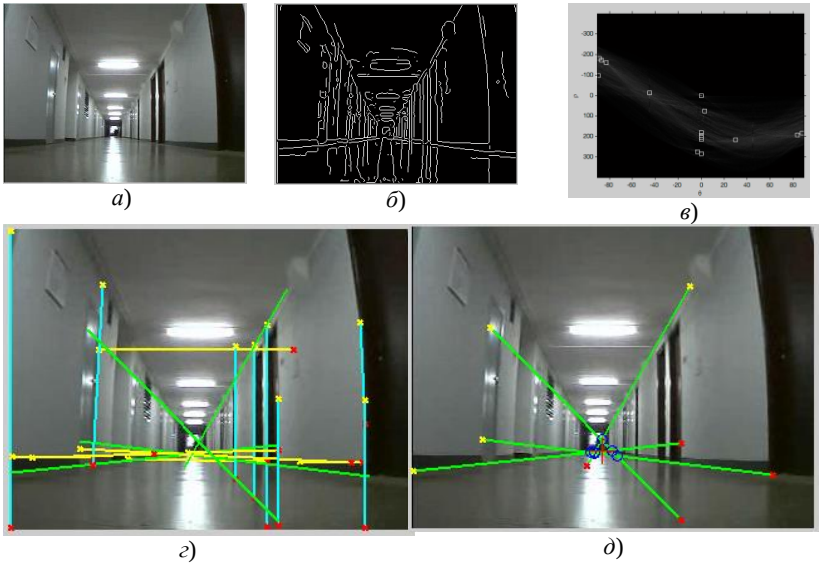



Рис. 3. Основные функциональные блоки системы технического зрения.

- блок Find All Lines From Hough Transforms соответствует блоку «Сегментация» (рис. 1*в*). Этот блок необходим для перевода контуров монохромного изображения в прямые отрезки и прямые линии с помощью применения стандартного преобразования Хафа (рис. 4*в*);
- блок Line Sort соответствует блоку «Классификация/кластеризация» (рис. 1*г*) и предназначен для сортировки всего набора отрезков на 3 класса: вертикальные, горизонтальные и диагональные. С помощью полученных данных были просчитаны реальные координаты найденных отрезков на рассматриваемом изображении (рис. 4*г*);
- блок Lines Crossing Points соответствует блоку «Принятие решений» (рис. 1*д*). Он предназначен для обнаружения точки перспективы и дальнейшего принятия решения корректировки движения АМР. Для решения поставленной задачи данный блок был настроен на пропускание только диагональных отрезков, т.к. на анализе их расположения и взаимного пересечения можно определить точку линейной перспективы. Другим словами, в блоке осуществляется сепарация и удаление всех вертикальных (голубых) и горизонтальных (жёлтых) отрезков, чтобы остались только диагональные отрезки (зелёные), с помощью которых можно найти координаты точки линейной перспективы (рис. 4*д*). Для выявления точки перспективы требуется найти все точки пересечения диагональных отрезков в кадре (синие кружки) и вычислить центр масс облака этих точек (красный крестик);



*а*) *б*) *в*)

*г*) *д*)

Рис. 4 Иллюстрация этапов выявления точки линейной перспективы

- На основе найденных координат расположения вычисленной точки линейной перспективы в кадре относительно оси *Y* в блоке Motion Direction, выходные значения которого который подаются на нижний уровень системы управления, принимается решение, в какую сторону АМР следует корректировать траекторию своего движения.

## 3.2. Simulink-модель управления нижним уровнем

На рис. 5 изображены основные функциональные блоки Simulink-модели, реализующей нижний уровень управления.

Блок Communication позволяет обмениваться микрокомпьютеру RPi2 и микроконтроллерной плате Arduino Mega управляющими сигналами и сенсорными данными по протоколу RS232. Для этого используется разработанный ранее командный протокол [Сычёв и др., 2014].

Блок PID Controller реализует ПИД-регуляторы по двум контурам управления: по скорости движения и углу поворота шасси АМР. Управляющие значения Left Wheel Speed, Right Wheel Speed и Direction, необходимые для функционирования следующего блока, определяются на основе вычисленных и фактических значений инкрементных энкодеров. Коэффициенты ПИД-регуляторов определялись эмпирически.

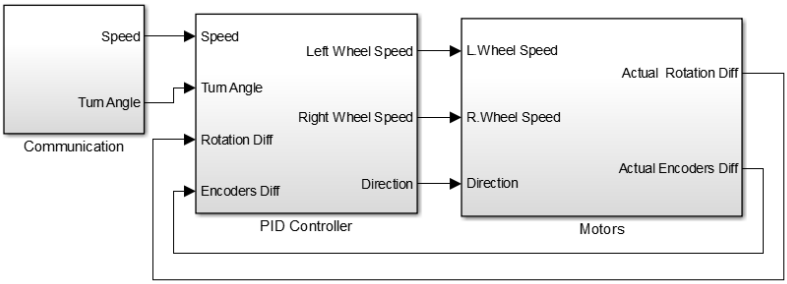



Рис. 5. Основные функциональные блоки системы управления нижнего уровня

Блок Motors реализует управление всеми четырьмя двигателями на основе переданных значений. Подсчёт количества тиков с каждого из энкодеров производится с помощью программных счётчиков.

## Заключение

Предложенная СТЗ была протестирована на разработанном прототипе АМР с двухуровневой системой управления в реальных условиях. Разработанные и реализованные алгоритмы управления позволили АМР безошибочно двигаться по середине коридоров трёх различных зданий с максимальной для выбранного шасси скоростью – 54 м/мин.

Таким образом, полученные результаты дают основание полагать, что полученные алгоритмы могут лечь в основу подпрограмм управления сервисными роботами, предназначенных для погрузочно-разгрузочных работ. Это могут быть роботы-санитары, развозящие еду по палатам больницы, либо транспортные роботы для работы на автономных товарных складах.

В дальнейших работах планируется повышать уровень робастности предложенного алгоритма пространственного ориентирования АМР, так как он был испытан в «пустых» коридорах, т.е. без наличия движущихся людей. Также следует уделить внимание алгоритмам поиска точки перспективы, если она будет в процессе движения утеряна либо не будет находится в момент старта.

## Список литературы

**[Герасюто и др., 2014]** Герасюто С.Л., Прокопович Г.А., Сычёв В.А. Построение навигационной карты внутри помещений по величине магнитного поля земли MEMS сенсором мобильного робота // Робототехника и техническая кибернетика. – 2014. – №3. – С. 53-56.

**[Прокопович и др., 2014]** Прокопович Г.А., Шагов Д.Н. Прототип автономного

транспортно-складского робота с развитой сенсорной системой // Автоматизация и роботизация процессов и производств : мат. респ. науч.-практич. семинара / редкол.: Пантелеенко Ф.И. (гл. ред.) [и др.]. – Минск : Бизнесофсет, 2014. – 2014. – С. 98-100.

**[Форсайт и др., 2004]** Форсайт Д.А., Понс Ж. Компьютерное зрение. Современный подход. // Вильямс. – 2004 г. – 928 с.

**[Гонсалес и др., 2006]** Гонсалес, Р., Р. Вудс, С. Эддинс. Цифровая обработка изображений в среде MATLAB / Гонсалес, Р., Р. Вудс, С. Эддинс // Техносфера. – 2006 г. – С. 410-421.

**[Герман-Галкин, 2008]** Герман-Галкин С.Г. Matlab & Simulink. Проектирование мехатронных систем на ПК. – СПб. : КОРОНА-Век. – 2008. – 368 с.

**[Сычёв и др., 2014]** Сычёв В.А., Герасюто С.Л. Разработка протокола и программно-аппаратного обеспечения системы централизованного управления группой роботов // Автоматизация и роботизация процессов и производств : материалы республиканского научно-практического семинара / редкол.: Пантелеенко Ф.И. (гл. ред.) [и др.] Минск : Бизнесофсет. –2014. С. 101-102.

**УДК 004.896:621.865**

# ОБ ОДНОМ МЕТОДЕ РАСПОЗНАВАНИЯ ОБЪЕКТОВ С НЕ ПОЛНОСТЬЮ ОПРЕДЕЛЕННЫМИ ПРИЗНАКАМИ

А.Д. Московский (*moscowskyad@gmail.com*)
НИЦ «Курчатовский институт»

**Аннотация**. В работе описывается комплексный метод распознавания объектов на изображении в условиях неполноты входных данных, для задач навигации мобильных роботов. Представлен разработанный программный модуль для ROS, представлены результаты экспериментов, эмулированных в среде Gazebo[1].

**Ключевые слова**: мобильная робототехника, распознавание, техническое зрение, реконструкция сцен, недоопределенные модели, доопределение атрибутов.

## Введение

Современная мобильная робототехника не обходится без задач технического зрения. Это связано, как с доступностью таких сенсоров, как камеры, так и с огромным потоком полезной информации, предоставляемым этими сенсорами. Разработано много методов, позволяющие распознавать разные элементы окружающего пространства. Среди всех можно выделить несколько основных подходов. Первая группа методов занимается извлечением из изображений трехмерных данных. Изначально это были подходы, рассматривающие стерео зрение [Horn и др., 1988], а затем так называемое SfM (Structure-from-Motion – Строение из Движения) [Huang и др., 1989]. Последний подход сосредотачивается на сравнении «близких» друг к другу изображений, сделанных во время движения камеры. Похожий подход активно был использован в целом классе навигационных алгоритмов SLAM (simultaneous localization and mapping – одновременная локализация и картирование) [Davison и др., 2007] и [Engel и др., 2014]. Также в подобным методам относится метод извлечения трехмерных данных по изображениям отснятым с разных точек наблюдения [Nistér, 2004]. Другая большая группа методов сосредоточена на детектировании объектов в окружающем пространстве.

[1] Работа выполнена при частичной поддержке гранта РФФИ 15-07-07483

Данные методы стараются ответить на вопросы «что?» и «где?» [Fallis, 2013]. Среди них тоже можно выделить несколько основных групп методов, одна занимается вопросами распознавания а классификации, кластеризации и категоризации объектов [Dickinson и др., 2010]. Больший интерес для данной работы представляют методы, выделяющие непосредственно объекты на изображении, большая группа посвящена методам, разбивающие объект на отдельные геометрические примитивы и детектирующие их, например алгоритмы RBC (Recognition-by-Components) [Biederman, 1985] и ACRONYM [Zisserman и др., 1995], данные подходы были созданы в попытке воссоздать человеческий подход к распознавание в техническом зрении. Очень актуальной в своё время являлась задача распознавания лиц, породившая такие методы, как распознавание образов [Turk и др., 1991] и каскадами Хаара [Viola и др., 2001]. Еще одна наиболее интересная группа методов занимающаяся выделением ключевых точек на изображении при помощи дифференциальных и градиентных методов [Lowe, 2004]. Наиболее известными и широко распространёнными методами являются алгоритмы SIFT [Lowe, 1999] и SURF [Bay и др., 2006]. Помимо выделения ключевых точек, существуют и алгоритмы выделения ключевых областей [Matas и др., 2004].

Все перечисленные методы обычно работают по принципу детектирования, т.е. либо находят данных объект на изображении, либо утверждают, что его нет. Решение об отсутствии объекта может быть вынесено ложно, например, по причине плохой видимости искомого объекта. Факторами, мешающими распознаванию, могут быть шум, перекрытие другими объектами, необычное освещение, неудачный ракурс наблюдения и др. Иными словами входная информация неполная (или недоопределенная). Однако методы работы с изображениями в условиях неопределенности достаточно редки, или специфичны для какого-то определенного класса объектов. Цель данной работы – предложить механизм обнаружения объектов на изображении (используя в основе существующие подходы к распознаванию), с помощью которого можно было бы выдвигать предположения о нахождении объектов, в условиях неопределённости.

## 1. Постановка задачи

Для многих задач мобильной робототехники была бы полезна информация о возможном нахождении искомого объекта, нераспознанного из-за неполноты наблюдаемых данных. Для того, что бы детектировать объекты в условиях недоопределённости, предлагается рассматривать каждый объект как совокупность неких признаков, которые в свою

очередь, могут быть отдельно от объекта быть распознаны неким существующим методом распознавания. Признаки должны подбираться таким образом, что бы частично видимый объект можно было потенциально (т.е. с некоторой оценкой уверенности) идентифицировать. Конечно, такой подход не сможет гарантировать нахождение объекта в случае его частичной видимости, однако должен позволить сделать заключение вида «возможно требуемый объект здесь» и сопоставить с этим некую числовую оценку уверенности. При таком подходе, однако, будут и отрицательные эффекты: увеличатся случаи ложного распознавания. Избежать этой проблемы можно расширяя информацию об объекте путем добавления информации не визуального характера (раз речь идет о мобильной робототехнике, то это всевозможные дополнительные сенсоры), а также понятие «контекста» объекта. Дальнейшие действия зависят от глобальной задачи робота, в каких-то случаях придется «поверить» методу, в каких-то «захватить» объект и следить за ним, до тех пор, пока точность распознавания не увеличится. В дальнейшем этот процесс будет описан как доопределение атрибутов.

# 2. Описание подхода

## 2.1 Объекты и атрибуты

Метод распознавания работает с понятием объекта. Объектом называется некая сущность, которую требуется распознать посредством технического зрения. Обычно объект отождествляется с предметами в окружении робота, например «стул», «чашка», «дверь» и т.п. Распознать объект означает выделить на получаемом от камеры изображении область, соответствующую реальному положению объекта в пространстве.

Как говорилось, идея метода «разложить» объект на общие для всех объектов составляющие. Данные составляющие совершенно естественно описываются признаками объекта, или же атрибутами. В качестве атрибутов объекта выступают такие понятия как форма, цвет и узор (текстура), а также ряд пространственных атрибутов, таких как отношение линейных размеров и положение в пространстве. Здесь приведен рабочий перечень атрибутов, расширять данный список можно какими угодно признаками, например, добавить анализ динамики объекта (подвижный\неподвижный) или рассматривать тени, отбрасываемые объектами. Каждый атрибут принадлежит своему пространству. Например, атрибут цвета может характеризоваться точкой или областью в трехмерном пространстве RGB или HSV, в то время как форма может принимать бесконечно много значений, и поэтому пространство представляется списком элементов, внесенных в каждую конкретную

реализацию. Для распознавания отдельных атрибутов, могут использоваться различные методы распознавания. Также следует отметить, что одни атрибуты могут лучше характеризовать объект, чем другие, для этого введем коэффициент значимости для каждого атрибута. Таким образом, каждый объект характеризуется набором пар атрибут-коэффициент:

$$O_i = \{(a_1, k_1) \dots, (a_N, k_N)\},\ a_j \epsilon A_j,\ k_j \epsilon \mathbb{R}, \quad (1)$$

где $O_i$ – объект, $a_i$ – значение атрибута из подпространства атрибутов $A_i$, $k_i$ – соответствующий атрибуту коэффициент из действительных чисел.

Однозначно распознанным в рамках данного подхода объект называется таким, у которого все атрибуты были распознаны в одной и той же области изображения. При этом, чем больший набор атрибутов будет у объекта, тем сложнее будет спутать этот объект с другими.

## 2.1 Комплексные объекты и сцены

Однако подобное описание объекта не удовлетворяет всем нуждам задач в робототехнике. Некоторые объекты не подвергаются подобному описанию, т.к. содержат несколько разных значений одного и того же атрибута, например объект многоцветный. Для этого было введено понятие комплексного объекта, представляющего собой набор простых объектов и отношений между ними. Отношениями называются связи между объектами, в основном используются пространственные отношения [Варосян и др., 1982], а также отношения линейных размеров. Такое расширение позволяет описать больший спектр объектов. И при этом есть возможность также объединять отношениями комплексные объекты, такие объединения согласно [Minsky, 1975] будем называть сценами.

## 2.2. Проблема частичной видимости

Подходя к проблеме частичной видимости, возникают два случая. Рассмотрим случай, когда частично виден простой объект. В данном случае могут распознаться в одной области не все атрибуты, например форму не удалось распознать, т.к. она закрывается другим объектом, а цвет наоборот удалось, как и примерный размер, не смотря на перекрытие. Два из трех атрибутов совпадет, далее введем коэффициент уверенности *CC* определяемый как отношение суммы коэффициентов распознанных атрибутов, к сумме всех атрибутов объекта.

$$CC_o = \frac{\sum_{detected} k}{\sum_{total} k}, \quad (2)$$

где, $k$ – коэффициенты атрибутов объекта из формулы 1.

Коэффициент уверенности принимает значения от нуля до единицы,

соответственно, если сила равна единице, объект считается однозначно распознанным.

Второй случай описывает ситуацию, когда комплексный объект или сцена частично видимы. В данном случае один или несколько объектов могут быть находиться в частичной видимости или отсутствовать вовсе. В таком случае коэффициент уверенности комплексного объекта или сцены определятся как отношение суммы по всем связям полусумм коэффициентов уверенности каждого объекта к общему количеству связей:

$$CC_s = \frac{\Sigma_{ratio}\, 0.5(CC_{Oj}+CC_{Oi})}{N_{ratio}}, \quad (3)$$

где $CC_{oi}$ – коэффициенты уверенности простых объектов, рассчитанных по формуле 2, $N_{ratio}$ – общее количество связей в комплексном объекте. Аналогично простому объекту коэффициент уверенности принимает значения от нуля до единицы и равен единице в том случае, когда все составляющие объекты однозначно распознаны, как и состоят в заданных отношениях друг с другом.

## 2.3 Недоопределенные модели

Как можно было заметить, объект представляет собой элемент много размерного пространства, причем каждая размерность может характеризоваться разными подпространствами, как было сказано раньше подпространство цвета – трехмерное, подпространство формы – набор дискретных значений. Однако для описания каждого атрибута требуется прибегнуть к унифицированному механизму. В роли такого механизма выступают недоопределенные модели (далее Н-модели) Нариньяни [Narin'yani и др., 1997]. Согласно парадигме Н-моделей объект на стадии распознавания является недоопределенной переменной, т.е. заданной не точно, а каким то интервалом. Данный интервал должен сужаться на каждом следующем шаге работы алгоритма, стремясь к точному значению.

Т.к. данные методы предполагается использовать в задачах мобильной робототехники, то не исключается возможность улучшения условий наблюдения на последующих шагах функционирования робота. Иными словами методы детектирования можно применять только к областям (или их расширениям), показавшим положительную силу на предыдущем шаге, при этом коэффициент у верности каждого следующего детектирования должен быть не хуже предыдущего.

$$f_t : D_{image} = D_t, \; CC_{Dt} > 0 \quad (4)$$

$$f_{t+1}: D_t = D_{t+1}, \; CC_{Dt+1} \geq CC_{Dt}, \quad (5)$$

где $f_t$ – функция детектирования на шаге $t$, $D$ – область изображения. Такой подход назовем доопределением атрибутов.

## 3. Эксперименты

Чтобы проверить применимость данного метода была создана система распознавания, включающая в себя детекторы для определения следующих атрибутов:

- Попиксельный детектор цвета.
- Гистограммный детектор цвета.
- Детектор формы каскадами Хаара [Viola и др., 2001].
- Детектор контуров Кэнни [Canny, 1986].
- Детектор линейных размеров объектов.

Также система способна распознавать пространственные отношения между объектами. Система написана на языке C++ под ROS (http://wiki.ros.org/indigo). Архитектура построена таким образом, что база детекторов легко расширяема. Эксперименты проводилась на модели, построенной в среде Gazebo (http://gazebosim.org). В качестве распознаваемых объектов были использованы двухкомпонентные цветные маркеры (как пример комплексного объекта) и морские навигационные знаки сторон света - вехи.

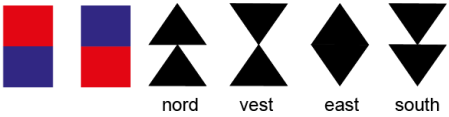



Рис. 1. Распознаваемые объекты

Окружение представляло собой модель проходного холла с дверьми колонами и текстурным полом.

На рисунке 2 представлено распознавание вехи «восток» (см. рис. 1). Слева дан рисунок с коэффициентом уверенности ограниченным снизу значением 0.6, справа - значением 0.8.

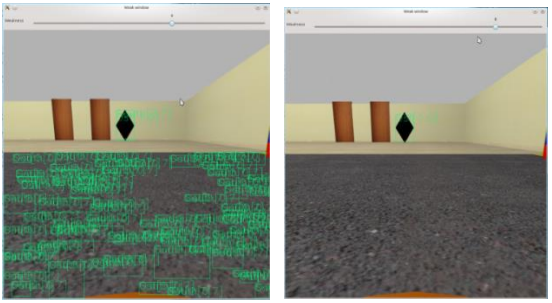


Рис. 2. Результат распознавания вехи «восток» при ограничении снизу коэффициента уверенности 0.6 (слева) и 0.8 (справа)

Описание вехи «восток» состояло из четырех атрибутов: характерная ромбовидная форма, черный цвет, продолговатый в вертикальной оси силуэт и средний размер (предполагается, что робот не должен видеть объект близко или вдали). Как видно из рисунка, метод определяет текстуру пола как искомые объекты. Коэффициент доверия у этих ложных объектов сильно ниже, чем у верно распознанного знака. Также эти объекты носили «случайный» характер и являлись ошибкой применения цветовой гистограммы к изображению. Это удалось исправить, введя еще один атрибут – повторяемость объекта. Данный атрибут давал свой вклад, только если на предыдущем шаге алгоритма этот же объект был обнаружен примерно в этой же области и с похожими другими атрибутами. Добавление нового атрибута увеличило способность распознавать объекты данного типа, уменьшив коэффициент доверия ложных объектов примерно на 0,15. Текстурный пол дает большое количество ложных объектов, на рис. 3 показан сравнительный график, поэтому важно ограничивать для каждого объекта нижний порог распознавания.

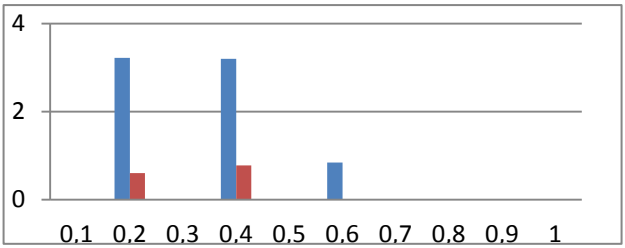



Рис. 3. Сравнительная гистограмма наличия ложных объектов при видимом текстурном полу (синий ряд) и при его отсутствии (красный ряд). По оси абсцисс отложен коэффициент уверености, по оси ординат логарифм колличества ложных объектов на изображеннии

На рисунке 4 представлена ситуация с частичным перекрытием распознаваемого объекта элементами окружения. Веха «восток» оказалась наполовину закрыта колонной, и распознать ее форму каскадом Хаара не удалось. Тем не менее, система предположила, что данный объект может находиться в данной области. Однако аналогичное представление было сделано касательно вехи «юг» также представленной на рисунке, это следствие того, что у этих двух объектов различимый атрибут только форма.

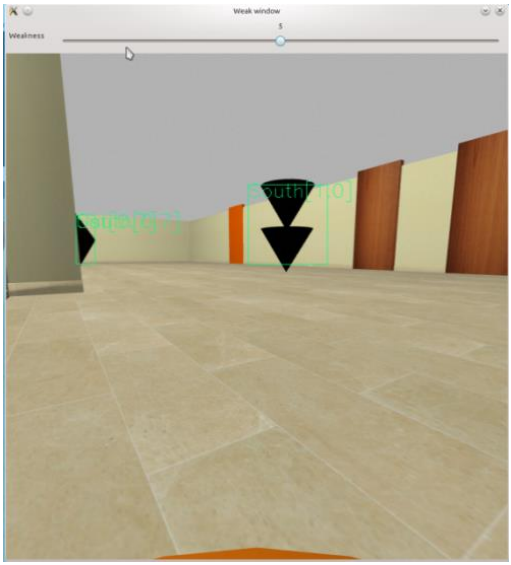



Рис. 4. Частичная видимость вехи «восток»

Аналогичная ситуация представлена на рисунке 5. Двухкомпонентный маркер состоит из двух простых объектов, соединенных пространственным отношением «над». Каждый составляющий объект является прямоугольник определённого цвета. На рисунке слева в окне представлены найденные методом простые объекты, справа – комплексные объекты. В этом случае, комплексный объект распознается с коэффициентом уверенности 0.5, согласно формуле 3.

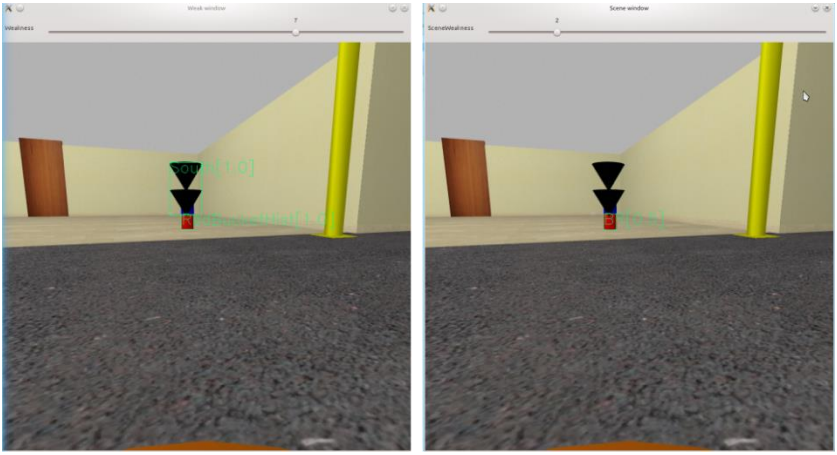


Рис. 5. Перекрытие комплексного объекта

Данные эксперименты показали потенциальную применимость метода для задач распознавания, однако робастность распознавания сильно зависит от того, какими атрибутами описывается объект и какие методы распознавания заложены в конкретной реализации. Также сильное влияние имеет выбор порогового значения для коэффициента доверия. На данный момент настройка данный параметров может осуществляться только вручную.

Дальнейшие эксперименты будут направлены на работу с более сложными объектами и на расширение базы распознавательных методов, заложенных в систему. Также планируется интеграция системы с другими системами мобильного робота, в частности с системой навигации, для получения «контекста» и использование его в качестве дополнительной информации при расчете коэффициентов уверенности найденных объектов.

## Заключение

Был разработан и протестирован на лабораторных объектах комплексный метод распознавания объектов. Объект в рамках метода представляется в виде набора атрибутов, распознаваемых отдельными методами. По количеству и композиции распознанных атрибутов делаются предположения о возможном нахождении объекта, даже в условиях частичной видимости. Приведенные эксперименты показали возможность применимости метода в задачах робототехники, в т. ч. в задаче навигации [Московский, 2015].

## Список литературы

**[Narin'yani и др., 1997]** A.S.Narin'yani, S.B.Borde, D.A.Ivanov. Sub-Definite Mathematics and Novel Scheduling Technology Programs // Artif. Intell. Eng. 1997. T. 11.

**[Bay и др., 2006]** Bay H., Tuytelaars T., Gool L. Van. SURF: Speeded up robust features // Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). , 2006. С. 404–417.

**[Biederman, 1985]** Biederman I. Human image understanding: Recent research and a theory // Comput. Vis. Graph. Image Process. 1985. Т. 32. С. 29–73.

**[Canny, 1986]** Canny J. A Computational Approach to Edge Detection // IEEE Trans. Pattern Anal. Mach. Intell. 1986. Т. PAMI-8. № 6. С. 679–698.

**[Davison и др., 2007]** Davison A.J. и др. MonoSLAM : Real-Time Single Camera SLAM // IEEE Trans. Pattern Anal. Mach. Intell. 2007. Т. 29. № 6. С. 1–16.

**[Dickinson и др., 2010]** Dickinson S.J. и др. Object Categorization, Computer and Human Vision Perspectives // J. Electron. Imaging. 2010. Т. 19. № 3. С. 039901.

**[Engel и др., 2014]** Engel J., Sch T., Cremers D. LSD-SLAM: Large-Scale Direct Monocular SLAM // Eccv. 2014. С. 1–16.

**[Fallis, 2013]** Fallis A.. Springer Handbook of Robotics. , 2013. 1689-1699 с.

**[Horn и др., 1988]** Horn B.K.P., Hilden H.M., Negahdaripour S. Closed-form solution of absolute orientation using orthonormal matrices // J. Opt. Soc. Am. A. 1988. Т. 5. № 7. С. 1127.

**[Huang и др., 1989]** Huang T.S., Faugeras O.D. Some Properties of the E Matrix in Two-View Motion Estimation // IEEE Trans. Pattern Anal. Mach. Intell. 1989. Т. 11. № 12. С. 1310–1312.

**[Lowe, 1999]** Lowe D.G. Object recognition from local scale-invariant features // Proceedings of the Seventh IEEE International Conference on Computer Vision. , 1999. С. 1150–1157.

**[Lowe, 2004]** Lowe D.G. Distinctive image features from scale-invariant keypoints // Int. J. Comput. Vis. 2004. Т. 60. № 2. С. 91–110.

**[Minsky, 1975]** Marvin Minsky. A Framework for Representing Knowledge // Psychol. Comput. Vis. 1975.

**[Matas и др., 2004]** Matas J. и др. Robust wide-baseline stereo from maximally stable extremal regions // Image and Vision Computing. , 2004. С. 761–767.

**[Nistér, 2004]** Nistér D. An efficient solution to the five-point relative pose problem // IEEE Trans. Pattern Anal. Mach. Intell. 2004. Т. 26. № 6. С. 756–770.

**[Turk и др., 1991]** Turk M., Pentland A. Eigenfaces for Recognition // J. Cogn. Neurosci. 1991. Т. 3. № 1. С. 71–86.

**[Viola и др., 2001]** Viola P., Jones M. Rapid object detection using a boosted cascade of simple features // Comput. Vis. Pattern Recognit. 2001. Т. 1. С. I–511–I–518.

**[Zisserman и др., 1995]** Zisserman a. и др. Class-based grouping in perspective images // Proc. IEEE Int. Conf. Comput. Vis. 1995. С. 183–188.

**[Варосян и др., 1982]** Варосян С.О., Поспелов Д.А. Неметрическая пространственная логика // Техническая кибернетика. 1982. Т. №5. С. 86–99.

**[Московский, 2015]** Московский А.Д. Система навигации автономного мобильного робота на основе метода реконструкции сцен // Робототехника и техническая кибернетика. 2015. Т. 4. № 9. С. с.47–55.

**УДК 629.7.021.6**

# АЛГОРИТМ УПРАВЛЕНИЯ БЕСПИЛОТНЫМ ЛЕТАТЕЛЬНЫМ АППАРАТОМ ТИПА КОНВЕРТОПЛАН

С.Ф. Яцун, О.В. Емельянова, К.Г. Казарян
(*teormeh@inbox.ru*)
ГОУ ВО "Юго-Западный государственный университет",
Курск, Россия

**Аннотация**. В статье рассматривается управляемое движение беспилотного летательного аппарата (БПЛА) вертикального взлета и посадки, типа конвертоплан, с центрально расположенным управляемым приводом, причем поворотным является гондола с винтами и двигателями. Предложен алгоритм системы автоматического управления конвертопланом в основных режимах движения: взлете, горизонтальном полете и посадке. Представляемая работа выполнена в рамках проекта РФФИ "Изучение закономерностей движения прыгающе-летающего робота".[1]

**Ключевые слова**: конвертоплан, математическое моделирование, система управления.

## Введение

Стремительное развитие малоразмерных высокоманевренных беспилотных летательных аппаратов (БПЛА) со множеством структурных и конструктивных решений, в последнее время, связанно с быстро растущими технологиями и инженерными решениями в различных областях промышленности [Jatsun et al, 2014], [Silva et al, 2010], [Ryll et al, 2012]. Они достаточно простоты в изготовлении; благодаря применению современных композитных материалов, имеют надежную и легкую конструкцию, и как следствие малую массу при существенной массе полезной нагрузки. Эти аппараты компактны и маневренны при реализации сложных алгоритмов управления [Рэндал и др 2014], [Павловский и др., 2014].

Одним из перспективных летательных аппаратов можно считать

[1] Работа выполнена при финансовой поддержке РФФИ (проект № 16-08-00787).

конвертоплан, в котором сочетаются признаки вертолетной схемы – вертикальный подъём (спуск) и самолетной схемы – полёт в горизонтальной плоскости при наличии крыла и за счёт изменения вектора тяги [Павловский и др., 2015]. Преимуществом такого аппарата является снижение энергопотребления бортовой системой, которое может достигаться за счёт применения летающего крыла, позволяющего совершать перелеты на дальние расстояния. В тоже время, в известных на сегодняшний день трехроторных схемах конвертопланов, при повороте одного (заднего) или двух боковых поворотных винтов, для перехода из вертикального полёта в горизонтальный, имеет место потеря высоты. Это не является критичным, если полет проходит на достаточных высотах и потеря высоты может быть быстро восстановлена.

Однако существует ряд задач, при выполнении которых необходимо обеспечить полет аппарата на сверхмалых высотах, порядка нескольких метров. В этом случае отсутствие запаса по высоте, в режимах, связанных с изменением направления тяговых сил, может приводить к контакту аппарата с землей в результате потери высоты. Поэтому, нужны новые схемы конвертопланов, летающих на таких режимах.

Предложенная в статье схема БПЛА с центрально расположенным управляемым приводом, центр масс которого совпадает с центром масс всей системы, лишена такого недостатка, так как поддержание заданной высоты полета обеспечивается силами тяги периферийных неповоротных приводов [Яцун и др., 2015а], [Яцун и др., 2014].

Для исследования основных режимов движения предложенной схемы ЛА необходимо построить алгоритм выработки управляющих воздействий; разработать систему управления движением и провести тестирование программного комплекса и инструментальных средств проектирования на основе пространственной модели конвертоплана.

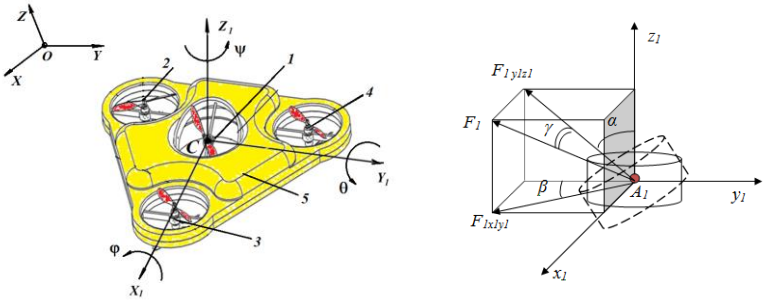
## 1 Описание и принцип работы конвертоплана

Рассмотрим модель конвертоплана с центрально расположенным управляемым приводом, который движется в неподвижной декартовой системе координат $OXYZ$, тогда $CX_1Y_1Z_1$ – подвижные системы координат, проходящие через центр масс корпуса аппарата $C$ (рис. 1, *а*). Ориентацию в пространстве задают самолетные углы рысканья $\psi$, тангажа $\theta$ и крена $\varphi$ [Павловский и др 1990].

Аппарат состоит из четырех пар управляемых винтов *1-4* на основе бесколлекторных электроприводов и крыла *5*, установленного на несущей раме, на которой также закреплены блок питания, плата управления и приёмник сигнала, электрически связанные с приводами вращения винтов. Причем, центрально расположенный электропривод *1* – с

изменяемым (относительно осей $CX_1Y_1Z_1$,) вектором тяги, винты *2, 3, 4* – с неизменяемыми векторами тяги. Отклонение вектора тяги поворотного привода *1* сначала в плоскости $A_1y_1z_1$ на угол $\alpha$, затем в плоскости $A_1x_1z_1$ на угол $\gamma$ (рис. 1, *б*), позволяет управлять полётом аппарата по заданной траектории. Компенсация крутящего момента в представленной несимметричной схеме происходит с помощью пар винтов, вращающихся в противоположных направлениях, (по два мотора на одной оси).

Такая схема является устойчивой к внешним воздействиям, отличается простотой исполнения, поможет обеспечить автономный полёт по заложенному маршруту. Для улучшения аэродинамических характеристик аппарата, снижения энергопотребления, данный аппарат снабжен летающим крылом, которое позволяет совершать горизонтальное перемещение с малым энергопотреблением [Яцун и др., 2015a], [Яцун и др., 2015b] [Яцун и др., 2016].




*а)* *б)*

Рис.1. Расчетная схема конвертоплана: *а* ─общий вид; *б*─ центральный поворотный привод

## 2. Система автоматического управления

В задачах управления БПЛА существует необходимость расчета положения объекта в пространстве, в заданные дискретные моменты времени. Формируемые команды управления содержат соответствующие алгоритмы для указанного маневра, данные о его начале и окончании. Так как конвертоплан совершает сложные перемещения в пространстве, то для упрощения бортовых вычислительных операций, необходимо разбивать их на последовательность более простых поступательных и вращательных движений. А использование специальных алгоритмов, позволит выполнять быстрые преобразования этих последовательностей на аппаратном уровне.

Существует много подходов к решению поставленной задачи, которые реализуются под каждый конкретный случай. Наиболее близкий подход изложен в работе [Хачумов, 2014], где движение БПЛА по заданному маршруту решается как задача преследования цели, основанная на эвристическом методе решения задачи коммивояжера (метод Кохонена). В данном методе используется принцип двухмерного отображения траектория движения распределения *n* опорных точек (нейронов-кластеров) на одномерный кольцевой маршрут, предварительно заполненный вспомогательными точками. После завершения обучения нейронной сети, положение кластера в маршруте определяется положением его образа в кольцевом выходном слое. Однако в работе предложено осуществлять движение объекта по маршруту с управлением углом тангажа и скоростью. Как показывает опыт, такой подход приводит к накоплению со временем погрешностей, и отклонениям следования от цели.

Данное обстоятельство привело к созданию собственного алгоритма системы управления, использующего некоторые модификации нейронной сети Кохонена. В работе предлагается определять движение объекта с помощью координат в инерциальной системе отчета. Достоинством предлагаемого алгоритма является простота реализации и хорошая результативность движения аппарата на заранее известной траектории с возвратом в исходную точку при условии минимального координатного отклонения. Недостатком является необходимость подгонки алгоритмов полета под конкретную, заранее неизвестную, траекторию.

Управление приводами предложенной конструкции конвертоплана генерируется бортовой системой управления по соответствующим алгоритмам. В состав системы управления входят компаратор, логический регулятор, объект управления и обратные связи [Atsushi Oosedo et al, 2015], [Ryll et al, 2012], [Рэндал и др 2014].

Управляющее напряжение питания от микроконтроллера, через бесколлектроные драйверы, подается на каждый из двигателей винтов *1*, *2, 3, 4* и на два сервопривода винта *1*, используемые для изменения вектора тяги поворотного винта, и определяющие углы $\alpha$, $\gamma$ оси вращения центрального винта. В качестве инерциальной навигационной системы используются: акселерометр, магнитометр, гироскоп. Магнитометр определяет значение угла поворота $\psi$; акселерометр - углы поворота $\varphi$, $\theta$; гироскоп показывает значение угловых скоростей $\omega_{XI}$, $\omega_{YI}$. Разрабатываемые нами алгоритм предполагают совместное комплексирование этих значений, что позволяет обеспечивать достаточно точную ориентацию конвертоплана в пространстве.

Информационная система конвертоплана включает в себя также GPS навигацию и 3-х координатный дальномер, необходимые для определения

координат центра масс аппарата *X, Y, Z* в инерциальной системе отсчета.

Полученные данные приходят на микроконтроллер где сравниваются с заданными значениями параметров, которые сохранены в постоянном запоминающим устройством (ПЗУ). Величина ошибки ($\Delta_i$) по соответствующим обобщенным координатам и углам, определяемая разностью между фактическими и заданными значениями, поступает на вход регулятора, который вычисляет значения управляющих напряжений $U_1$–$U_6$, подаваемых на электродвигатели, в соответствии с принятой стратегией $U_i=U_i\,(\Delta_i)$.

Таким образом, система управления обеспечивает движение конвертоплана по заданной траектории [Яцун и др., 2016].

Величину вектора ошибки $\bar{\Delta}(t)$ в реальном масштабе времени *t* можно представить как:

$$\bar{\Delta}(t) = \bar{q}^*(t) - \bar{q}(t),$$

где $\bar{q}^*(t) = \begin{vmatrix} X^* \\ Y^* \\ Z^* \\ \varphi^* \\ \theta^* \\ \psi^* \end{vmatrix}$, $\bar{q}(t) = \begin{vmatrix} X \\ Y \\ Z \\ \varphi \\ \theta \\ \psi \end{vmatrix}$ -заданное и полученное значения обобщенных координат и углов.

Обозначим значение ошибки в начальный момент времени ($t$=0), как $\bar{\Delta}_0$, а её первую производную как $\bar{\upsilon}_0$, тогда

$$\bar{\Delta}(0) = \bar{\Delta}_0, \qquad \frac{d}{dt}\bar{\Delta}(0) = \bar{\upsilon}_0.$$

Асимптотическое стремление ошибки к нулю будет обеспечено, если ошибка удовлетворяет уравнению:

$$\frac{d^2}{dt^2}\bar{\Delta} + k_d\frac{d}{dt}\bar{\Delta} + k_p\bar{\Delta} = 0,$$

где $k_p$, $k_d$ — коэффициенты усиления пропорциональной и дифференциальной составляющих регулятора соответственно:

$$k_p = \begin{vmatrix} k_{1X} & 0 & 0 & 0 & 0 & 0 \\ 0 & k_{1Y} & 0 & 0 & 0 & 0 \\ 0 & 0 & k_{1Z} & 0 & 0 & 0 \\ 0 & 0 & 0 & k_{1\varphi} & 0 & 0 \\ 0 & 0 & 0 & 0 & k_{1\vartheta} & 0 \\ 0 & 0 & 0 & 0 & & k_{1\psi} \end{vmatrix}, \quad k_d = \begin{vmatrix} k_{2X} & 0 & 0 & 0 & 0 & 0 \\ 0 & k_{2Y} & 0 & 0 & 0 & 0 \\ 0 & 0 & k_{2Z} & 0 & 0 & 0 \\ 0 & 0 & 0 & k_{2\varphi} & 0 & 0 \\ 0 & 0 & 0 & 0 & k_{2\vartheta} & 0 \\ 0 & 0 & 0 & 0 & & k_{2\psi} \end{vmatrix}.$$

Характеристические уравнения имеют вид:

$$\lambda_i^2 + k_{i+3}\lambda_i + k_i = 0,$$

а их решения: $\Delta_i = C_{1,i}e^{\lambda_{1,j}t} + C_{2,i}e^{\lambda_{2,j}t}, \quad i = 1-3,$

где $C_{i,j}$ – константы интегрирования, определяемые из начальных условий, $\lambda_{i,j}$ – корни $j$-го характеристического уравнения ($j$=1…6).

Все корни отрицательные и действительные, при условии, что элементы матрицы $k_p$, $k_d$ удовлетворяют следующим соотношениям:

$$4k_{1X} < k_{2X}^2; \quad 4k_{1Y} < k_{2Y}^2; \quad 4k_{1Z} < k_{2X}^2; \quad 4k_{1X\varphi} < k_{2\varphi}^2; \quad 4k_{1\vartheta} < k_{2\vartheta}^2; \quad 4k_{1\psi} < k_{2\psi}^2.$$

Такой подход позволяет определять область параметров многомерного PD-регулятора, обеспечивающего оптимальную работу системы управления конвертопланом. Использование интегрального коэффициента, как показывает практика, на таком аппарате не предполагается вследствие достаточной инерционности конвертоплана.

## 3. Синтез алгоритма управления движения конвертоплана по траектории

Траектория движения конвертоплана будет состоять из трех этапов: вертикальный взлет, полет в горизонтальной плоскости и посадка [Яцун и др., 2015b], [Яцун и др., 2016].

Рассмотрим подробнее каждый из этапов.

1 этап– взлёт: $0 \le Z \le H$, $X=X_0=R$, $Y=Y_0$, $Z=Z(t)$, где $X_0, Y_0$ – параметры взлёта, $H$ - высота подъёма; $\alpha=0^0$, $\gamma=0$. Тяговые силы $F_i=b\omega_i^2$, $i=1\text{-}4$, параллельны оси $CZ_1$ и формируются в зависимости от ошибки по высоте: $F_i = F_i$ ($\Delta Z$), где $b$–аэродинамическая составляющая. Начало полета конвертоплана происходит в момент времени, когда нормальная реакция опорной поверхности $\bar{N}=0$; при этом выполняется условие: $Mg < F1z+F2+F3+F4$.

2 этап – полет по заданному закону (рис.2). Пусть траектория движения конвертоплана в плоскости $XOY$ задана окружностью, радиуса $R$: $Z=H$, $X= R\cos(\Omega t)$, $Y=R\sin(\Omega t)$, где $R$ – радиус зоны облета, $\Omega R$ –

максимальная скорость движения центра масс конвертоплана по траектории. Уравнение траектории имеет вид: $X^2 + Y^2 = R^2$.

Для обеспечения движения конвертоплана по траектории развернем поворотный винт на угол $\alpha=90^0$ в направлении движения и управляемыми параметрами будут являться: - $135^0 \le \gamma=\beta \le 135^0$, $\beta=\beta(\Delta\psi)$ – формируется в зависимости от ошибки по углу рысканья; силы тяги неповоротных винтов *2-4* формируются также как на 1 этапе: $F_i=F_i$ $(\Delta Z)$, $i=2\text{-}4$, а поворотного винта $F_1 = F_1$ $(\Delta X, \Delta Y)$ – в зависимости от ошибок по перемещению.

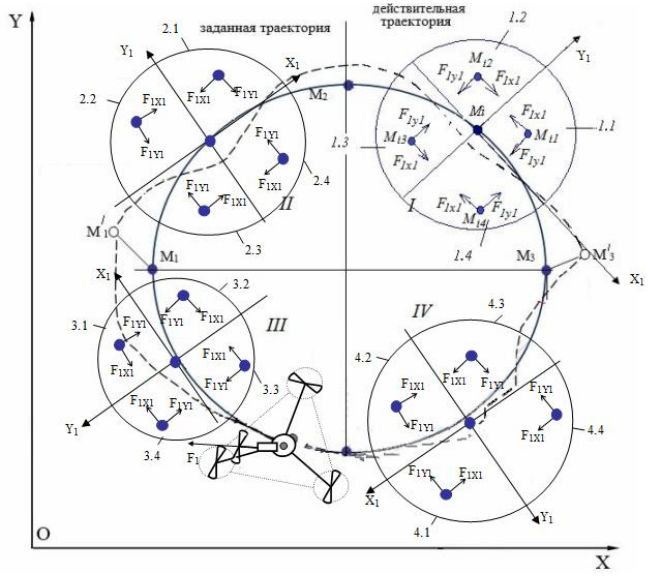



Рис.2. Схема алгоритма управления движением конвертоплпнп по заданной траектории

Для составления алгоритма полета конвертоплана в горизонтальной плоскости зададимся направлением зоны облета – против хода часовой стрелки. Условно разделим окружность на 4 четверти и контроль за действительным положением объекта $M_i^{/}$ $(i=1\ldots n)$ будем осуществлять с помощью совокупностей точек $M_i$, расположенных в каждой из четвертей. Для построения системы управления, удовлетворяющей заданной траектории, будем фиксировать координаты, изменение координат, получаемые от объекта по отношению к точкам, находящейся в конкретной четверти, определять ошибки перемещения $\Delta x$ и $\Delta y$, где $\Delta X_i = X_i^* - X_i$ - погрешность перемещения по оси $X$; $\Delta Y_i = Y_i^* - Y_i$ -

погрешность перемещения по оси *Y*.

Для минимизации величины ошибки необходимо настроить систему управления таким образом, чтобы угол β поворотного центрального винта, относительно осей подвижной системы координат $X_1Y_1Z_1$, а соответственно и знаки проекций тяговой силы $F_{1X_1}, F_{1Y_1}$ обеспечивали минимальное отклонение центра масс конвертоплана от заданной траектории в пространстве.

Рассмотрим последовательность наведения конвертоплана на точки $M_i$, алгоритм управления движением представляет собой решение задачи по определению в каждой четверти траектории требуемой скорости и управляющих сил, обеспечивающих асимптотическое сближение действительного и заданного положения аппарата. В каждой четверти можно выделить 4 зоны, в которых определяем действительное положение точки $M_i^{/}$, знак проекции силы $F_1$ на оси подвижной системы координат и значение угла β, тангенс угла которого, определен как: $\operatorname{tg}\beta = \frac{F_{1Y_1}}{F_{1X_1}}$.

Например: для *I* четверти (фрагмент на рис. 2):

точка $M_{i1}^{/}$ (зона 1.1): $F_{1X_1} = -F_{1X_1}$; $F_{1Y_1} = -F_{1Y_1}$; $\beta = 45^0$;

точка $M_{i2}^{/}$ (зона 1.2): $F_{1X_1} = F_{1X_1}$; $F_{1Y_1} = -F_{1Y_1}$; $\beta = -45^0$;

точка $M_{i3}^{/}$ (зона 1.3): $F_{1X_1} = F_{1X_1}$; $F_{1Y_1} = F_{1Y_1}$; $\beta = -135^0$;

точка $M_{i4}^{/}$ (зона 1.4): $F_{1X_1} = -F_{1X_1}$; $F_{1Y_1} = F_{1Y_1}$; $\beta = 135^0$.

Знаки угла β указывают направление и величину, на которую необходимо повернуть проекцию силы тяги поворотного винта $F_1$, чтобы обеспечить заданную траекторию движения.

3 этап – спуск и посадка: $0 \le Z \le H$, $X=X_0=R$, $Y=Y_0$, $Z=Z(t)$. Данный этап аналогичен первому этапу. Здесь $\alpha=0^0$, $\gamma=0^0$. Тяговые силы $F_i=F_i(\Delta Z)$, $i=1\text{-}4$, параллельны оси $CZ_1$ и являются функциями ошибки по высоте.

Результаты математического моделирования представлены на рис. 3. На рисунке 3 представлены зависимости изменения проекций сил тяги управляемых пар винтов с неизменяемыми векторами тяги ( $\bar{F}_2 - \bar{F}_4$) и поворотного винта *1* ($F_{1x}$ и $F_{1y}$) на каждом из этапов полета: взлете (*I*), горизонтальном полете (*II*) и посадке(*III*). По ним можно судить о характере движения ЛА в пространстве и путем изменения углов α, γ поворотного винта изменять направление движения в горизонтальной плоскости, а так же контролировать заданную высоту подъёма конвертоплана.

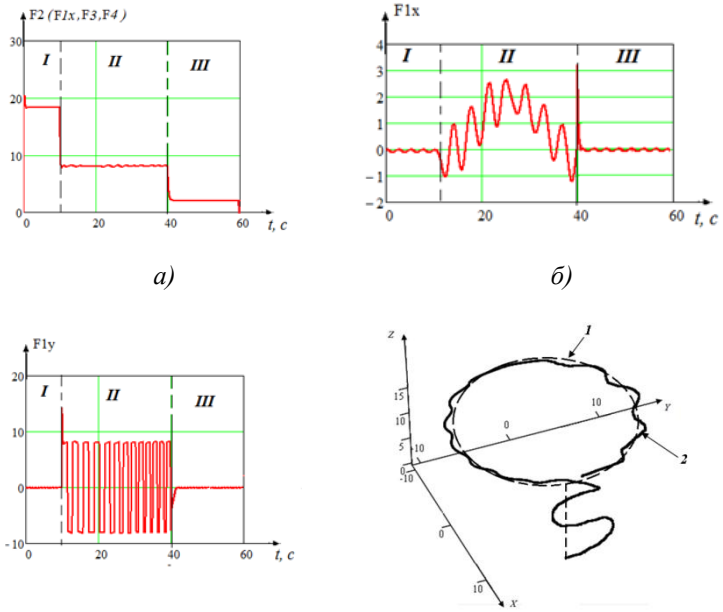



Рис.3. Графики зависимостей проекций тяговых сил от времени: *а* – на ось *Z*, *б* – на ось *X*, *в* – на ось *Y; г* - пространственная фактическая траектория движения; *1*- заданная траектория, *2* – фактическая траекторияот времени; *I, II,III* – этапы полета

Анализ графиков показывает, что использование пропорционально-дифференциального регулятора позволяет обеспечить достаточно устойчивое движение конвертоплана по заданной траектории. Данный метод управления может быть использован при наличии достаточно точной информации о пространственной траектории исследуемого объекта. Результаты математического моделирования показывают устойчивость системы управления.

## Выводы

Предложена конструкция и расчетная схема конвертоплана с центрально расположенным управляемым приводом, алгоритм выработки управляющих воздействий, который заключается в использовании области рациональных параметров коэффициентов для ПД регулятора, определяющих работу приводов аппарата в различных режимах полета по

заданной траектории.

Предложен метод управления движением аппарата в горизонтальной плоскости, включающий задание произвольной траектории, позволяющий минимизировать ошибки перемещения летательного аппарата.

## Список литературы


**[Atsushi Oosedo et al, 2015]** Atsushi Oosedo, Satoko Abiko, Shota Narasaki, Atsushi Kuno, Atsushi Konno, Masaru Uchiyama. Flight control systems of a quad tilt rotor unmanned aerial vehicle for a large attitude change. Robotics and Automation (ICRA), 2015 IEEE International Conference on. –P. 2326-2331.

**[Jatsun et al, 2014 ]** Jatsun S.F. Mathematical model of the quadrotor type unmanned aerial vehicle with neurocontrollerl. / S.F. Jatsyn, V.E. Pavlovsky, O.V. Emelyanova, A.S. Savitsky // Advances in Robotics, Mechatronics and Circuits: proceedings of the 18th International Conference on Circuits (CSCC'14) and proceedings of the 2014 International Conference on Mechatronics and Robotics, Structural Analysis (MEROSTA 2014), Santorini Island, Greece. –2014. - P. 46-50.

**[Silva et al, 2010]** Silva C., Yeo, H., and Johnson, W. Design of a Slowed-Rotor Compound Helicopter for Future Joint Service Missions. Proceedings of the American Helicopter Society Aeromechanics Specialist's Conference, San Francisco, CA, January 20-22, 2010.

**[Ryll et al, 2012]** M. Ryll, H. H. Bulthoff and P. Robuffo Giordano. Modeling and Control of a Quadrotor UAV with Tilting Propeller. Robotics and Automation (ICRA), 2012 IEEE International Conference on. PP. 4606-4613.

**[Павловский и др., 2014]** Павловский В.Е. Моделирование и исследование процессов управления квадрокоптером / В.Е. Павловский, С.Ф. Яцун, О.В. Емельянова, А.В. Савицкий // Робототехника и техническая кибернетика: научно-техн. журнал / Санкт-Петербург.–2014.– №4(5). –С.49-57.

**[Павловский и др., 2015]** Павловский В. Е. Математическое моделирование робота с переменным вектором тяги/ В. Е. Павловский, С. Ф. Яцун, О. В. Емельянова, С. П. Стуканёва // Второй всероссийский научно-практический семинар «Беспилотные транспортные средства с элементами искусственного интеллекта» (БТС-ИИ-2015)»: Труды семинара. – Санкт –Петербург: Изд-во «Политехника-сервис», –2015. –С.99-106.

**[Павловский и др 1990]** Павловский М.А., Акинфеев Л.Ю., Бойчук О.Ф. Теоретическая механика. Динамика.-К.: Выща шк., 1990.-480 с.

**[Рэндал и др 2014]** Рэндал У.Биард. Малые беспилотные летательные аппараты: теория и практика/ Рэндал У.Биард, Тимоти У. МакЛейн // Москва:ТЕХНОСФЕРА, 2015.-312 с.

**[Хачумов, 2014]** Хачумов М.В. Задача облета беспилотным летательным аппаратом с учетом ветровых нагрузок/М.В. Хачумов// Второй всероссийский научно-практический семинар «Беспилотные транспортные средства с элементами искусственного интеллекта» (БТС-ИИ-2015)»: Труды семинара. – Санкт –Петербург: Изд-во «Политехника-сервис», –2015. –С.130-135.

**[Яцун и др., 2014]** Яцун С. Ф. Моделирование движения квадрокоптера при

отклонении осей несущих винтов относительно корпуса / С. Ф. Яцун, О.В. Емельянова, А.И. Савин // Вибрационные технологии, мехатроника и управляемые машины: сб. науч. ст.: в 2ч. Ч.1/ Юго-Зап. гос. ун-т. Курск. –2014.–С.329- 338.

**[Яцун и др., 2015a]** Яцун С.Ф. Моделирование полёта конвертоплана в различных режимах движения / С. Ф. Яцун, О. В.Емельянова, А. И. Савин, С. П.Стуканёва //Известия ЮЗГУ. Серия техника и технологии. 2015. - №1(14) – С. 55-66.

**[Яцун и др., 2015b]** Яцун С.Ф. Математическое моделирование конвертоплана с центрально расположенным управляемым приводом./ С. Ф. Яцун, О. В. Емельянова, А. И. Савин // Известия ЮЗГУ. Серия техника и технологии. –2015. - №4(17) – С. 31-37.

**[Яцун и др., 2016]** Яцун С.Ф. Алгоритм управления движением конвертоплана с центральным управляемым вектором тяги / С.Ф. Яцун, О.В. Емельянова, Г.В.Климов, С.П.Стуканева// Вибрационные технологии, мехатроника и управляемые машины: сб. науч. ст.: в 2ч. Ч.1/ Юго-Зап. гос. ун-т. Курск, 2016.–С.50- 62.

**УДК 004.932.2**

# МНОГОПОТОЧНАЯ РЕАЛИЗАЦИЯ АЛГОРИТМА ВИЗУАЛЬНОЙ ЛОКАЛИЗАЦИИ БПЛА НА ОСНОВЕ ИЗВЕСТНОЙ 3D МОДЕЛИ ОКРУЖЕНИЯ И ТЕХНОЛОГИИ CUDA

А.К. Буйвал (*alexbuyval@gmail.com*)
М.А. Гавриленков (gavrilenkov@umlab.ru)
Брянский государственный технический университет, Брянск

**Аннотация**. В статье приводится описание реализации с использованием технологии CUDA алгоритма визуальной локализации БПЛА внутри помещения на основе сопоставления граней, полученных из изображения с видеокамеры и из смоделированного изображения, полученного на основе известной 3D модели окружающей среды (помещения). В конце статьи дается сравнение производительности последовательной и многопоточной реализации на основе моделирования в среде Gazebo.

**Ключевые слова**: визуальная локализация, фильтр частиц, ROS, Gazebo

## Введение

В настоящий момент всё еще остается открытым вопрос о быстрой и точной локализации мобильного робота и в частности БПЛА в пространстве. Локализация мобильного робота в пространстве является нетривиальным процессом, требует детального рассмотрения во многих аспектах [Pink, 2009]. В связи с тем, что локализация в закрытом пространстве не может осуществляться по таким параметрам, как координаты GPS, на практике применяются либо дорогие лазерные дальномеры, либо более продвинутый подход – использование информации с бортовой камеры [Saurer, 2010] и фильтра частиц. Данный подход основывается на сравнении изображения, полученного с камеры, со смоделированным изображением, полученным на основе 3D модели помещения. Данный подход во многом превосходит альтернативные решения, но, к сожалению, не лишен недостатков. Весомым недостатком становится избыточность получаемой с камеры информации.

Избыточность информации ведет к большей затрате ресурсов на очищение изображения, нахождения ключевых точек и граней и т.д. К

сожалению, на сегодняшний момент не существует универсального решения этой проблемы. Дополнительным негативным фактором является ограниченность вычислительных ресурсов на борту БПЛА. Одним из выходов из сложившейся ситуации является использование альтернативных вычислительных мощностей на борту БПЛА.

В данной статье предлагается рассмотреть идею частичного использования GPU в системе локализации БПЛА для наиболее вычислительно сложных операций. В статье представлено описание реализации вычислительно сложных частей алгоритма, как на CPU, так и на GPU, а также приводится сравнение производительности при использовании этих двух подходов. Повышение производительности в работе алгоритма локализации также значительно повышает точность локализации, т.к. появляется возможность использования большего количества частиц.

# 1 Описание базового алгоритма локализации

Базовым элементов разработанной системы визуальной локализации является использование граней в изображении как характерных визуальных признаков. Ключевым моментом в таком подходе является сравнение и численная оценка схожести 2-х изображений: полученного с камеры и изображения, смоделированного на основе модели помещения [Nuske, 2009]. Предполагается, что алгоритм строит ряд гипотез о положении робота в пространстве, затем на основе этих гипотез он моделирует изображения, каждое из которых соответствует изображению, которое должен получить робот, если он находиться в этой предполагаемой точке. Таким образом, на каждой итерации алгоритма мы сравниваем одно изображение с камеры со множеством смоделированных изображений и определяем степень их схожести [Буйвал, 2015а].

## 1.1 Использование фильтра частиц для локализации робота

Для обработки данных о схожести изображений, а также данных с других доступных датчиков используется алгоритм локализации, основанный на множестве частиц (гипотез о местоположении робота) – фильтр частиц (particle filter). Данный метод широко известен и хорошо зарекомендовал себя в подобных задачах. Преимуществом этого метода является то, что он позволяет использовать множество гипотез, а также нелинейные модели как самой системы, так и нелинейные модели показаний датчиков [Буйвал, 2015а].

## 1.2. Измерительная модель фильтра частиц

Для того, чтобы оценить вероятность каждой гипотезы (частицы) необходимо оценить близость границ, полученных из изображения с камеры, к границам, полученным из смоделированного изображения, соответствующего гипотезе.

В разработанной системе используется метод ближайших граней. Данный метод не имеет такого недостатка, как учет только совпадающих на гранях пикселей. Т.о. если грани одного и того же конструктивного элемента на изображении с камеры и на смоделированном изображении не совпадают, но проходят близко друг к другу, то это все равно будет учитываться в конечной вероятности, т.к. в данном методе предлагается оценивать близость смоделированной грани к грани на изображении с камеры с помощью набора нормалей, построенных от смоделированной грани. Впервые данный метод был продемонстрирован [Nuske, 2009] и хорошо зарекомендовал себя для колесного робота.

В рассматриваемом методе выполняются последовательно следующие этапы:

1. Выделение границ на изображении, полученном с камеры и нахождение прямых среди них.
2. Рендеринг изображения согласно вектору состояния частицы, выделение границ на полученном изображении и нахождение прямых.
3. На каждой прямой формируется набор точек с постоянным шагом. Из каждой точки строится нормаль некой предельной длины. Если эта нормаль пересекается с какой-либо прямой из полученных на основе изображения с камеры, то длина такой нормали учитывается в общем весе рассматриваемой прямой по следующей формуле:

$$g(d) = exp\left(-\frac{d^2}{2\sigma^2}\right), \quad (1)$$

где $d$ – длина нормали, $\sigma$ – параметр определяющий вес нормали в зависимости от длины нормали (позволяет либо усиливать влияние длинных нормалей, либо уменьшать).

Для каждой прямой вычисляется общий вес путем суммирования всех весов нормалей по следующей формуле:

$$l = \frac{\sum_{i=0}^{S} g(d_i)}{S}, \quad (2)$$

где $S$ – общее количество прямых в смоделированном изображении.

Вычисляется итоговая вероятность гипотезы на основе объединения весов каждой линии по следующей формуле:

$$W = \propto \cdot exp\left(k\frac{\sum_{n=0}^{m} l_n}{m}\right), \quad (3)$$

где $m$ – количество прямых, $\propto, k$ – параметры учета веса прямых в

итоговом результате [Буйвал, 2015b].

## 2 Реализация фильтра частиц на CPU

Основным и, пожалуй, самым значимым моментом в работе фильтра частиц является обновление его весов. Как описано выше на первых этапах алгоритма необходимо выделить грани на изображении и найти прямые. Для этого используются стандартные функции пакета OpenCV: Canny и HoughLinesP соответственно. Количество частиц в системе задается статически и никак не корректируется.

На рис. 1. представлена схема вычисления весов частиц. Из схемы видно, что основной проблемой при классическом подходе к вычислению весов, является итеративность этих вычислений, при этом, само вычисление веса для частицы («Вычисление веса частицы») является финальным шагом внешнего цикла, и зависит от параметров, полученных во внутренних циклах. Сложность алгоритма в данном случае сравнима с N*K*M, где N-число частиц, K-число прямых на смоделированном изображении, M-число нормалей на прямой. Причем на каждой итерации внутреннего цикла нам необходимо произвести вычисления согласно формуле 1. Существует возможность распараллелить данный алгоритм в рамках использования CPU, но выигрыш в данном случае будет прямо пропорционален количеству задействованных процессоров. Стоит отметить, что мобильные роботы, зачастую, не имеют на своем борту процессоров с большим числом ядер, поэтому параллелизм в рамках CPU малоэффективен.

## 3 Реализация фильтра частиц на GPU

Проанализировав всё вышеописанное, напрашивается вывод, что для быстроты вычисления весов модели, лучшим решением является использование параллельных вычислений с большим числом ядер. Наиболее перспективной и мощной технологией таких вычислений автор видит технологии CUDA, разработанную компанией NVIDIA.

Работая с CUDA, программист получает параллельную вычислительную систему, работающую по принципу SIMT (Single-Instruction, Multiple-Thread), когда одна команда параллельно выполняется множеством независимых потоков (threads). Совокупность всех этих потоков, запущенных в рамках одной задачи (рис.2), носит название grid.[Habrahabr, 2012]

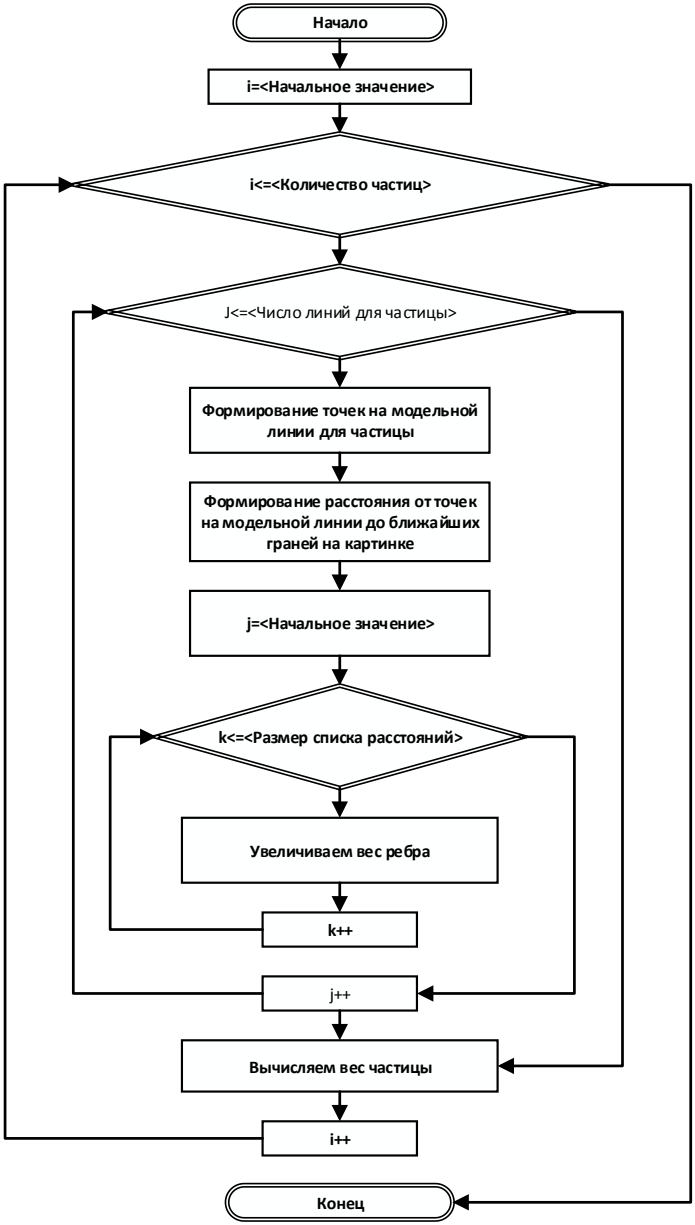



Рис. 1. Схема вычисления весов частиц с использованием CPU

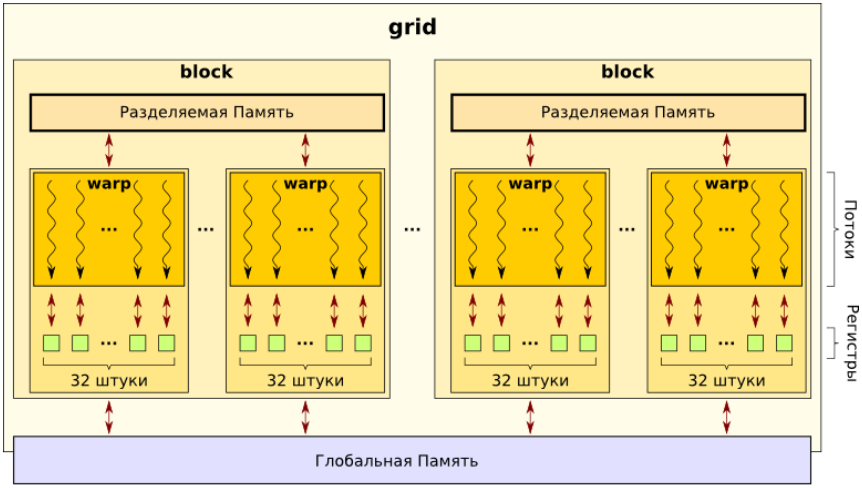



Рис 2. Иерархия потоков CUDA

В видеокарте, поддерживающий технологию CUDA, параллельное выполнение grid обеспечивается наличием в самой видеокарте большого количества скалярных процессоров (scalar processors). Данные процессоры и выполняют потоки (threads). Каждый скалярный процессор входит в более крупное образование (warp), тем самым являясь частью потокового мультипроцессора (streaming multiprocessor). Варпы также входят в другие крупные образования – блоки (blocks, рис. 2).

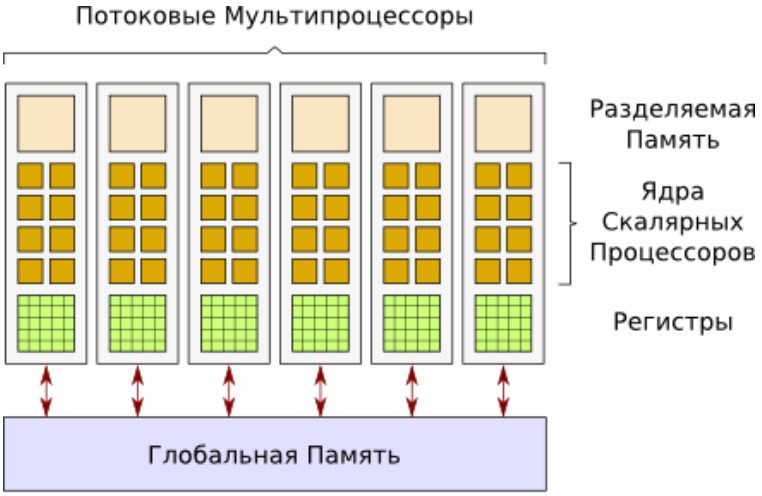



Рис. 3. Схема видеокарты с технологией CUDA

Другой, не менее важной особенностью, является организация памяти в CUDA и доступа потоков к её различным частям. Наивысшей степенью общедоступности для потоков обладает глобальная память (global memory), Расположение вне процессора делает этот тип памяти наиболее медленным, по сравнению с другими, предоставляемыми для вычислений на видеокарте.

Следующим типом в иерархии памяти идет разделяемая память (shared memory). Данный вид памяти расположен в каждом отдельном потоковом мультипроцессоре. Данная память доступна только тем потокам, которые выполняются на ядрах этого мультипроцессора. Так как к параллельному исполнению на одном мультипроцессоре может быть отведено более одного блока, то весь доступный в мультипроцессоре объем разделяемой памяти распределяется между этими блоками поровну. Необходимо упомянуть, что разделяемая память физически расположена где-то очень близко к ядрам мультипроцессора, поэтому обладает высокой скоростью доступа, сравнимой с быстродействием регистровой (registers) – основным видом памяти. Именно регистры могут служить операндами элементарных машинных команд, и являются более быстрой памятью [Habrahabr, 2012].

Несомненно, что при работе с несколькими потоками, следует использовать механизмы для синхронизации. CUDA предусматривает несколько команд для этих целей [Habrahabr, 2012]:

- **__syncthreads()** – самый верный способ. Эта функция заставит каждый поток ждать, пока все остальные потоки этого блока

достигнут этой точки или все операции по доступу к разделяемой и глобальной памяти, совершенные потоками этого блока, завершатся и станут видны потокам этого блока.

- **__threadfence_block()** будет заставлять ждать вызвавший её поток, пока все совершенные операции доступа к разделяемой и глобальной памяти завершатся и станут видны потокам этого блока.
- **__threadfence()** будет заставлять ждать вызвавший её поток, пока все совершенные операции доступа к разделяемой памяти станут видны потокам этого блока, а операции с глобальной памятью всем потокам на «устройстве». Под «устройством» понимается графический адаптер.
- **__threadfence_system()** подобна __threadfence(), но включает синхронизацию с потоками на CPU.

Исходя из всего вышеизложенного, была принята следующая структура алгоритма изменения весов фильтра частиц.

Как видно на схеме (рис. 4), вычисления с применением CUDA распараллеливают процесс работы вычисления весов частиц, что должно приводить к многократному ускорению этого процесса.

### 3.1. Дополнительные улучшения, основанные на использовании GPU

Как было описано выше, в предложенной подсистеме используются стандартные функции OpenCV [OpenCV, 2016a]. Основными используемыми функциями являются **Canny** [OpenCV, 2016b] и **HoughLinesP** [OpenCV, 2016c]. Данные методы выполняют довольно тяжеловесные операции по извлечению информации из изображения. Логичным ходом стал перенос вычислений из области вычисления процессора в область вычисления видеокарты [OpenCV, 2016e]. В статье [Luo, 2008] показана эффективность применения GPU для оператора Canny. В OpenCV существуют специальные альтернативные методы, которые способны выполняться на GPU: **cv::gpu::Canny** и **cv::gpu::HoughLinesP** [OpenCV, 2016d].

Процесс вызова GPU функций в таком случае выглядит следующим образом:

1. копирование исходные данных в память GPU;
2. вызов необходимых операторов OpenCV;
3. копирование полученных данных в память CPU.

## 4 Сравнительный анализ подходов

В ходе симуляционных экспериментов в среде ROS/Gazebo, был

проведен сравнительный анализ двух реализаций описанного алгоритма (только CPU и CPU+GPU), результаты которого приведены в таблице 1.

По результатам сравнения можно сделать вывод, что использование связки CPU+GPU дает огромный выигрыш в производительности. В среднем, в ходе эксперимента скорость выполнения алгоритма пересчета весов частиц увеличилась в среднем в 17,26 раз. Исходя из данных результатов можно сделать вывод, что данный подход является крайне эффективным.

Использование GPU для фильтра частиц является верным шагом к построению подсистемы локализации, главным достоинством которой является быстродействие.

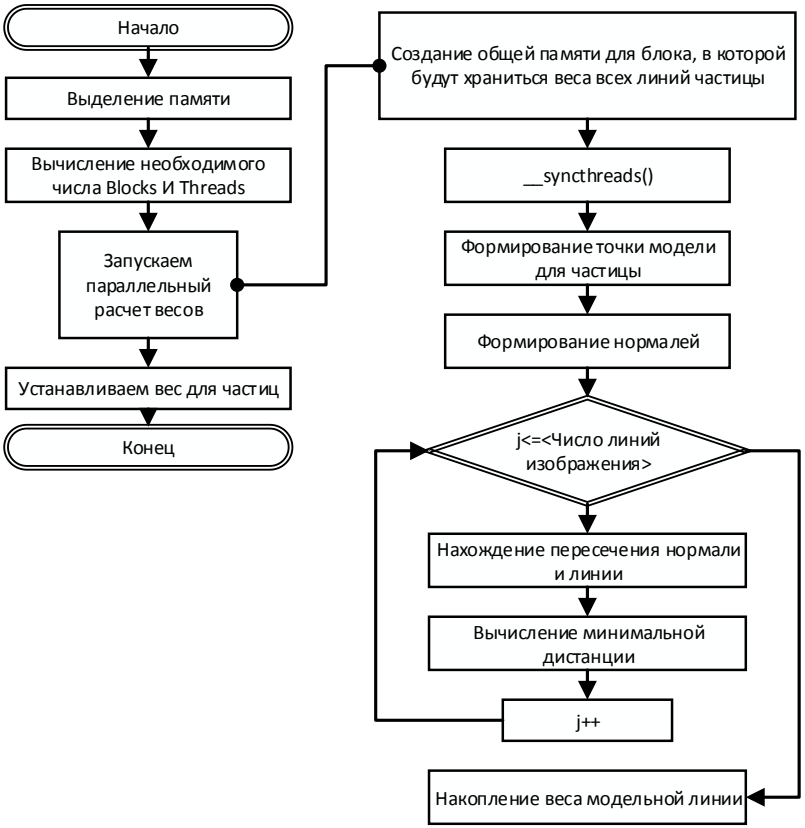



Рис. 4. Схема вычисления весов частиц с использованием GPU

Таблица 1.

| **Номер опыта** | **1** | **2** | **3** | **4** | **5** | **6** | **7** |
|---|---|---|---|---|---|---|---|
| **CPU time, мс** | 123,60 | 102,83 | 150,12 | 131,84 | 146,74 | 113,83 | 163,70 |
| **GPU time, мс** | 7,39 | 6,54 | 8,44 | 7,45 | 8,46 | 7,03 | 8,28 |
| **CPU/GPU, раз** | 16,7 | 15,7 | 17,7 | 17,6 | 17,3 | 16,1 | 19,7 |

## 5 Направления дальнейших исследований

В дальнейшей работе планируется испытать усовершенствованную в результате исследований систему на одноплатном микрокомпьютер **NDIVIA Jetson TX1**. Данный микрокомпьютер имеет встроенный GPU микропроцессор, что позволит получить существенный выигрыш во времени локализации при использовании разработанного алгоритма.

## 6 Заключение

В данной работе была модернизирована подсистема локализации БПЛА, а также было продемонстрировано преимущество использования GPU процессоров на борту БПЛА, были проанализированы стандартные функции библиотеки OpenCV для работы с CPU и GPU. В результате удалось добиться многократного ускорения работы алгоритмов локализации, что очень важно в условиях ограниченности ресурсов на борту БПЛА.

## Список литературы

**[Nuske, 2009]** Nuske, S., Roberts, J., & Wyeth, G. Robust outdoor visual localization using a three-dimensional-edge map // Journal of Field Robotics. 2009 №26(9), 728-756.

**[Буйвал, 2015a]** Буйвал А.К. Локализация беспилотного летательного аппарата внутри помещений на основе визуальных геометрических маркеров и известной 3D модели окружающей среды. //Ж-л Робототехника и техническая кибернетика. 2015 №3(8), 71-75 стр.

**[Буйвал, 2015b]** Буйвал А.К., Гавриленков М.А. Визуальная локализация на основе геометрических признаков и 3D модели окружающей среды в системе ROS.//Ж-л Робототехника и техническая кибернетика. 2015 №4(9), 56-59 стр.

**[Habrahabr, 2012]** Habrahabr. CUDA: синхронизация блоков. –https:// habrahabr.ru/post/151897.

**[OpenCV, 2016a]** OpenCV. – http://opencv.org.

**[OpenCV, 2016b]** OpenCV. Описание функции Canny. – http://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/canny_detector/canny_detector.html.

**[OpenCV, 2016c]** OpenCV. Описание функции HoughLines. –http://docs.opencv.org/2.4/modules/imgproc/doc/feature_detection.html?highlight=houghlines

**[OpenCV, 2016d]** OpenCV. Описание GPU функций Canny и HoughLines. – http://docs.opencv.org/2.4.13/modules/gpu/doc/image_processing.html.

**[OpenCV, 2016e]** OpenCV. Сравнительный анализ CPU и GPU. – http://opencv.org/platforms/cuda.html.

**[Luo, 2008]** Luo Y.,Duraiswami R. Canny Edge Detection on NVIDIA CUDA Perceptual Interfaces and Reality Laboratory Computer Science & UMIACS, University of Maryland, College Park. – http://www.umiacs.umd.edu/~ramani/pubs/luo_gpu_canny_fin_2008.pdf

**[Pink, 2009]** Pink, O., Moosmann, F., Bachmann, A. – Visual Features for Vehicle Localization and Ego-Motion Estimation. – http://www.mrt.kit.edu/z/publ/download/Pink_IV09.pdf

**[Saurer, 2010]** Saurer, O., Fraundorfer, F., Pollefeys, M. – Visual localization using global visual features and vanishing points. – http://people.inf.ethz.ch/saurero/publications/saurer_2010clef_visual_localization.pdf

**УДК 004.383.8.032.26**

# НЕЙРОМОРФНЫЙ ЧИП «АЛТАЙ», ОРИЕНТИРОВАННЫЙ НА ПРИМЕНЕНИЕ В СИСТЕМАХ ТЕХНИЧЕСКОГО ЗРЕНИЯ, РТК И БЕСПИЛОТНЫХ ТРАНСПОРТНЫХ СРЕДСТВАХ

В.М. Канглер (*vkangler@motivnt.ru*)
К.Е. Панченко (*kpanchenko@motivnt.ru*)
ООО “Мотив”, Новосибирск

**Аннотация**. Рассмотрен проект создания нейроморфного чипа не фон Неймановской архитектуры, моделирующего импульсные нейронные сети (spiking neural networks) и ориентированного на использование во встраиваемых системах, в системах технического зрения и управления робототехническими комплексами. Обозначены ключевые проблемы современных вычислительных систем и недостатки фон Неймановской архитектуры. Продемонстрирована неэффективность существующих вычислительных архитектур в решении ряда задач и обоснована необходимость новой вычислительной архитектуры, которая должна вдохновляться примером человеческого мозга. Показана перспективность использования нейроморфной архитектуры. Перечислены типы задач, которые эффективно решаются с помощью искусственных нейронных сетей. Выражена фактическая безальтернативность использования аппарата искусственных нейронных сетей в решении неформализуемых и слабо формализуемых задач. Представлен нейроморфный чип “Алтай” и обозначены его ключевые преимущества. Обоснованы выбранный поход к обучению моделируемых нейронных сетей, а также целесообразность цифровой реализации по сравнению с аналогово-цифровой. Описаны используемая модель нейрона, архитектурные принципы, структура всего чипа, а также структура и функционирование отдельного нейроядра. Приведен ряд моделей применения разрабатываемого чипа в робототехнических комплексах и системах специального назначения.

**Ключевые слова**: нейроморфные технологии; импульсные нейронные сети; архитектура фон Неймана; нейроморфная архитектура; аппаратная реализация нейронных сетей; нейроморфный чип.

## Введение

Наступающая новая эра интеллектуальных систем, интернета вещей, роботов и беспилотного транспорта нуждается в новых вычислительных средствах, способных в реальном режиме времени обрабатывать гетерогенные "зашумленные данные" большого объема. Для того чтобы эффективно решать задачи по распознаванию образов, глубокому анализу данных и автономному управлению, эти новые вычислительные системы должны сочетать в себе производительность суперкомпьютеров, низкое энергопотребление и высокую надежность. Из этого следует, что новые вычислительные системы должны иметь новую не фон Неймановскую высокопараллельную архитектуру. В природе такие системы уже существуют — это биологические нервные системы, самой продвинутой из которых является человеческий мозг.

## 1 Ограничения, накладываемые традиционной фон Неймановской архитектурой при реализации аппарата искусственных нейронных сетей (ИНС)

Наращивание количества ядер в микропроцессорах является основным способом повышения их производительности. Производители уже отказались от классической трактовки “закона Мура” ввиду увеличения сложности в одноядерных процессорах и интерпретируют этот закон по-новому — количество процессорных ядер на кристалле удваивается примерно каждые 18 месяцев. Если ориентироваться на такие формулировки, то в ближайшие десятилетия будут получены кристаллы с миллионами процессорных ядер. Хотя вопрос автоматического распределения кода по вычислителям все еще является “ненайденным Святым Граалем” и остается одним из ключевых вопросов вычислительных наук. В то же время биологический мозг демонстрирует высочайший уровень параллелизма и эффективности. Это и вдохновляет ученых и инженеров на создание нейроморфных устройств [Furber et al., 2013].

Традиционно аппарат искусственных нейронных сетей (ИНС) использовался для решения следующих задач: распознавание образов, реализация ассоциативной памяти, аппроксимация функций, управление процессами и автономными системами, фильтрация, сглаживание, прогнозирование. Разнообразие этих задач свидетельствует в пользу универсальности нейронных сетей как систем обработки информации [Хайкин, 2006]. В связи с развитием робототехники и увеличением потребности в быстрой обработке растущих объемов данных и мультимедиа информации растет и заинтересованность в эффективном

решении упомянутых задач. Это подстегивает интерес к системам, моделирующим большие нейронные сети. Моделирование нейронных сетей требует существенных вычислительных ресурсов. Для моделирования используются нейроускорители на сигнальных процессорах (DSP), графических процессорах (GPU) или ПЛИС (FPGA). Однако, параллельная и управляемая событиями природа нейронных сетей не является естественной для модели последовательной обработки традиционной компьютерной архитектуры (архитектура фон Неймана). В архитектуре фон Неймана необходима высокая пропускная способность для пересылки состояния входных и выходных значений нейронов между физически разделенными процессорами и памятью. Это приводит к высокому энергопотреблению и ограничивает возможности масштабирования таких систем [Imam et al., 2012]. Нейроморфные архитектуры изначально лишены этих недостатков и являются более естественными для моделирования нейронных сетей.

## 2 Актуальность применения нейроморфных технологий в системах технического зрения, навигации и управления

Актуальность нейроморфных технологий также подтверждается цитатой из книги вице-президента IBM по стратегическим решениям и исследованиям Джона Келли. "Архитектура фон Неймана так долго используется, потому что она обеспечивает мощное средство решения многих вычислительных задач. Большинство программ, написанных для сегодняшних компьютеров, основаны на этой архитектуре. Она имеет недостаток, который делает ее неэффективной — это так называемое "узкое горлышко архитектуры фон Неймана". В этой архитектуре каждый этап обработки требует нескольких шагов, где данные и команды перемещаются туда-сюда между памятью и процессором. Это требует огромного количества перемещений данных и обработки. Это также означает, что шаги обработки должны выполняться последовательно. И хотя здесь можно ввести некоторый параллелизм, но этого явно недостаточно. В течение последних десятилетий разработчики могли увеличивать возможности процессоров, делая их меньше и быстрее. В таком подходе мы почти достигли предела наших возможностей, и как раз в то время, когда мы нуждаемся в еще большей вычислительной мощности, чтобы справиться с требуемой сложностью и обработкой больших данных. Это приводит к невыполнимым требованиям для сегодняшних компьютерных технологий, в основном потому что современные компьютеры требуют колоссальной энергии для их выполнения.

Что необходимо, так это новая вычислительная архитектура, которая

должна вдохновляться примером человеческого мозга.

Обработка данных должна быть распределена по всей вычислительной системе, а не быть сосредоточенной в центральном процессоре. Обработка и память должны быть тесно интегрированы, чтобы уменьшить челночную пересылку данных и команд туда-сюда. И этапы обработки должны выполняться одновременно, а не последовательно. Когнитивный компьютер, используя такую архитектуру, будет реагировать на запросы быстрее, чем сегодняшние компьютеры, потребует меньшего количества перемещений данных; будет использоваться меньше энергии. Это не означает, что архитектура фон Неймана не будет использоваться. Она будет использоваться вместе с новой архитектурой в единых гибридных системах." [Kelly, 2013]

Под нейроморфными, как правило, понимают устройства аппаратно моделирующие работу импульсных нейронных сетей (Spiking Neural Networks, SNN).

Нейроморфные приборы будут востребованы в робототехнике: в системах технического зрения, навигации и управления. Развитие “традиционной” микроэлектроники позволило робототехнике (БПЛА, промышленные роботы, беспилотные транспортные средства) постоянно увеличивать функциональность и гибкость. Несмотря на это, интеллектуальные функции по-прежнему выполняются человеком — оператором. Обычно для этого требуются двунаправленные каналы связи с высокой пропускной способностью между оператором и роботом. Несмотря на то, что это приемлемо для многих ситуаций, повышение интеллектуальности и автономности управления за счет применения нейроморфных приборов кардинально изменит сценарии использования роботов в большинстве областей применения. Такие качества как энергоэффективность, функционирование в реальном режиме времени и размещение высокоуровневой системы управления на борту очень важны для робототехнических систем, в которых длительность автономной работы, удаленность от центра управления и короткое время реакции являются критичными [NICE, 2015]. Ярким примером таких систем могут быть космические робототехнические системы.

## 3 «АЛТАЙ» — разрабатываемый цифровой КМОП нейроморфный чип, моделирующий работу импульсных нейронных сетей

Разрабатываемый нейроморфный чип “Алтай” — «вычислительное» устройство, функционирующее на принципах схожих с биологическими нейронными системами.

В качестве основных моделей применения рассматриваются:

- Обработка изображений, в том числе кадров видеоряда в различных спектрах, например, в видимом, тепловом и других, с целью обнаружения, выделения и категоризации значимых объектов.
- Обработка акустических, в том числе и гидроакустических, сигналов с целью выделения интересующих акустических паттернов.
- Обработка физиологических сигналов. Например, анализ электроэнцефалограммы с целью организации эргономичного взаимодействия человека с робототехническими системами.

К ключевым преимуществам нейрочипа “Алтай” можно отнести:

- Высокая эффективность по энергопотреблению, производительности и размерам.
- Возможность решения неформализуемых и плохо формализуемых задач.
- Высокая масштабируемость, ограниченная только требованиями по энергопотреблению и массогабаритным параметрам.

## 3.1 Основополагающие принципы построения нейрочипа

“Алтай” должен обеспечивать эффективное функционирование произвольной импульсной нейронной сети.

Настройка параметров нейронов сети, в том числе и обучение, задание структуры сети, связей между нейронами, должны выполняться вне чипа с использованием инструментальных программных средств.

Вынесение задач обучения за пределы кристалла расширяет возможности выбора подходов и способов формирования параметров сети, с тем чтобы получить требуемую выходную функцию, зависящую от входных данных, как для отдельных подсетей, так и для всей моделируемой сети в целом. Также это повышает эффективность использования нейрочипа на этапе функционирования обученной сети.

## 3.2 Архитектура нейрочипа «Алтай»

### *3.2.1 Ключевые архитектурные решения*

В качестве основополагающих были выбраны следующие архитектурные решения:

- Полная цифровая реализация на КМОП технологии [Imam et al., 2012].
- Достаточно универсальная, но простая в реализации на кристалле цифровая модель нейрона. Модель должна поддерживать возможность выполнять широкий набор вычислительных

функций одним нейроном или группой нейронов: арифметические операции; логические операции; классическое поведение нейронов; обработку сигналов и вероятностное вычисление. [Cassidy et al., 2013].

- Параметры функционирования нейронов формируются вне кристалла [Arthur et al., 2012]. Это позволит использовать различные алгоритмы обучения нейронной сети. Кроме того, параметры нейронов могут быть получены не только в результате выполнения процедуры обучения, но и жестко заданы, если нужно реализовать на нейронной сети какую-то конкретную функцию, например, когда часть сети выполняет фильтрацию изображения функцией свертки.
- Многоядерная архитектура. Сам нейрочип представляет собой масштабируемую сеть нейроядер. Ядро — обособленная небольшая группа импульсных нейронов. Коммуникация спайков внутри ядра реализуется матрицей связей (crossbar), которая обеспечит высокую пропускную способность сигналов и исключит их взаимную блокировку. Сам нейрочип представляется как двумерная сеть из ядер. Для коммуникации между ядрами используются разделяемые шины и относительная AER (Address Event Representation) маршрутизация. Такой подход позволит масштабировать нейронные сети, объединяя множество нейрочипов в единый вычислительный комплекс.
- Нейрочип проектируется по модели GALS (Globally Asynchronous Locally Synchronous, глобально асинхронная локально синхронная схема) [Wang, 2008]. Ядра являются синхронными схемами, каждое из которых функционирует в своем домене синхронизации. Все коммуникационные блоки нейрочипа являются асинхронными. Такой подход является более сбалансированным по показателям энергопотребления, производительности и эффективности использования площади кристалла.

### *3.2.2 Используемая модель нейрона*

В проекте используется модификация “классической” модели нейрона — LIF (Leaky Integrate-and-Fire) с постоянной величиной утечки. Модель несколько упрощена, в ней используется только целочисленная арифметика.

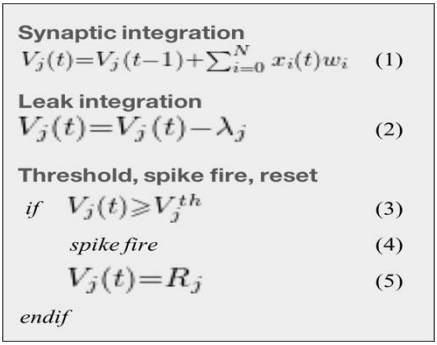



Рис. 1. Используемая модель нейрона.

Для *j*-го нейрона в момент времени *t* мембранный потенциал $V_j(t)$ является суммой потенциала на предыдущем шаге $V_j(t-1)$ и синаптических входов. Для каждого из *N* синапсов, синаптический вход равен произведению входного спайка на синапсе $x_i(t) \in \{0;1\}$ с его синаптическим весом $w_i$. Затем из накопленного потенциала вычитается величина утечки $\lambda_j$. Если накопленная величина $V_j(t)$ превышает порог $\alpha_j$, то нейрон испускает импульс и мембранный потенциал "сбрасывается" до значения $R_j$.

### *3.2.3 Архитектура нейрочипа и принцип функционирования*

Нейрочип "Алтай" включает в себя следующие функциональные блоки:

- Двумерную решетку нейроядер (32x32, 1024 ядра).
- 4 последовательных двунаправленных асинхронных порта ввода-вывода.
- Блок конфигуратора, предоставляющий I2C интерфейс для загрузки параметров моделируемой импульсной сети в нейроядра.
- Блок мониторинга, который позволяет контролировать работу нейрочипа.
- Блок автоматического регулирования внутренней частоты (ФАПЧ — фазовая автоподстройка частоты).

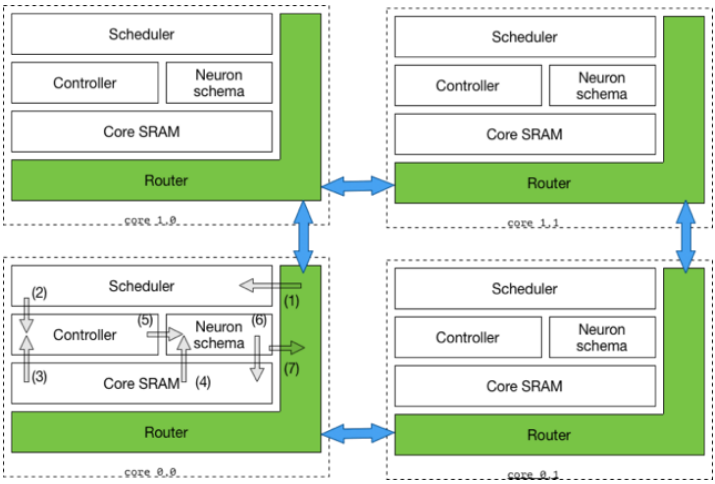



Рис. 2. Фрагмент решетки нейроядер.

Структурно ядро состоит из пяти функциональных блоков:

- Router — коммуницирует с соседними нейроядрами, принимая и маршрутизируя спайки.
- Scheduler — упорядочивает по времени и по входам (аксонам) приходящие спайки.
- Core SRAM — хранилище параметров функционирования нейронов ядра.
- Controller — реализует алгоритм управления последовательностью вычислений.
- Neuron schema — схема, реализующая логику функционирования нейрона.

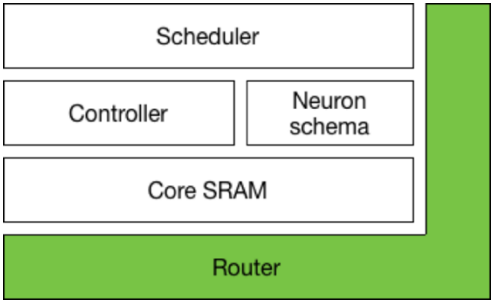



Рис. 3. Структура нейроядра.

Ядро моделирует $2^n$ нейронов и $2^m$ входов (аксонов).

Когда спайк достигает целевого ядра, маршрутизатор направляет его планировщику (1), где спайк буферизируется.

В каждый такт планировщик передает контроллеру все спайки, соответствующие по порядку текущему такту (2). Получив спайки, контролер организует последовательность вычислений для $\mathbf{2^n}$ нейронов по одному (один за другим).

В течение одного такта обсчитываются все $\mathbf{2^n}$ нейронов.

## 3.3 Оценки параметров энергопотребления и производительности

Проект по разработке нейроморфного чипа "Алтай" находится на начальной стадии и в данный момент проходит этап верификации HDL-кода нейроядер и всего чипа на “золотой модели”. Кроме того, получены оценки параметров энергопотребления для варианта, ориентированного на отечественный 90нм техпроцесс производства.

Ниже в таблице приведены планируемые параметры.

Таблица. 1. Планируемые характеристики нейрочипа «Алтай»

| | |
|---|---|
| Количество нейроядер на чип | 1024 |
| Количество моделируемых нейронов | 262144 |
| Количество синапсов | 67108864 |
| Производительность — количество синаптических операций в секунду (Sops) | $67*10^9$ |
| Период шага обработки (сек.) | $1*10^{-3}$ |
| Внутреняя частота работы нейроядер (МГц) | 70 |
| Напряжение питания (В) | 1 |
| Площадь кристала ($мм^2$) | 225 |
| Оценка мощности одного ядра (мВт) | 0.6 |
| Оценка потребляемой мощности (Вт) | 0.612 |

| | |
|---|---|
| Предполагаемый тип корпуса | TQFP-144<br>144 вывода<br>22мм x 22мм |

## 4 Сравнение с существующими решениями

Одной из самых очевидных и перспективных областей применения разрабатываемого чипа являются системы технического зрения. В настоящий момент эта область переживает революционные изменения. Системы, основанные на свёрточных глубоких нейронных сетях (CDNN), начиная с 2012 года, демонстрируют лучшие результаты в задачах распознавания образов и классификации изображений. В ряде типовых задач алгоритмы CDNN в качестве распознавания изображений сравнялись с человеком [LeCun et al., 2015], [Esser et al., 2016].

Сравним оценочные параметры разрабатываемого чипа с известными проектами на примере моделирования свёрточных нейронных сетей. Моделирование свёрточных сетей является вычислительно очень ресурсоёмкой задачей. Обучение CDNN требует миллионы циклов и по времени может занимать недели даже на мощных современных GPU. Поэтому в мире многие исследователи сосредоточены на оптимизации структур и алгоритмов функционирования CDNN, а также на том, чтобы предложить эффективные аппаратные решения, позволяющие использовать обученные CDNN во встраиваемых системах.

Для автономных и встраиваемых систем ключевыми параметрами являются производительность (количество кадров, обрабатываемых в секунду), эффективность (количество энергии необходимое для обработки одного кадра) и точность распознавания. Если сравнивать популярные микропроцессоры, графические процессоры, FPGA и ASIC (application-specific integrated circuit, интегральная схема специального назначения), то по этим параметрам решения, основанные на ASIC, вне конкуренции. Ниже приводится таблица отражающая параметры моделирования AlexNet [Krizhevskyet al., 2012] на общедоступных CPU и GPU. Видно, что GPU/mGPU значительно опережает CPU как по производительности, так и по энергоэффективности [Nvidia, 2015].

Таблица. 2. Основные параметры существующих конкурентных решений

| | Intel Xeon E5-2698 v3 | Intel Core i7-6700K | Nvidia Titan X | Nvidia Tegra X1 |
|---|---|---|---|---|

| Тип | CPU | CPU | GPU | mGPU |
|---|---|---|---|---|
| Призводительность (fps) | 476 | 242 | 3216 | 258 |
| Потребляемая мощность (W) | 149 | 62.5 | 227 | 5.7 |
| Энергоэффективность (J/f) | 0.313 | 0.258 | 0.071 | 0.022 |

Рассмотрим наши результаты в сравнении с последними достижениями схожей тематики. Конечно, напрямую сравнивать не совсем корректно, так как нужно сравнивать на одних и тех же задачах/тестах, но сравнение на близких задачах позволит понять хотя бы порядки величин. В таблице (#) приводим результат сравнения наших расчетных показателей с информацией из ряда последних публикаций [Nvidia, 2015], [Han et.al, 2016] , [Esser et al., 2016], [Chen et al., 2016].

Таблица. 2. Сравнения существующих решений с нейрочипом «Алтай»

| | Nvidia Titan X | Nvidia Tegra X1 | A-Eye | IBM True-North | MIT Eyeriss | Altai (our) |
|---|---|---|---|---|---|---|
| Тип | GPU | mGPU | FPGA | ASIC | ASIC | ASIC |
| Год | 2015 | 2015 | 2015 | 2016 | 2016 | 2016 |
| Призводитель-ность (fps) | 3216 | 258 | 8 | 1738 | 17 | 980 |
| Потребляемая мощность (W) | 227 | 5.7 | 9.63 | 0.208 | 0.1175 | 0.5212 |
| Энергоэф-фективность (J/f) | $7.1 * 10^{-2}$ | $2.2 * 10^{-2}$ | 1.203 | $1.2 * 10^{-4}$ | $6.9 * 10^{-3}$ | $5.3 * 10^{-4}$ |

Проведенный сравнительный анализ с существующими передовыми решениями аппаратного ускорения свёрточных сетей показал, что не смотря на то, что чип Алтай ориентирован не на самый современный технологический процесс производства (техпроцесс 90нм, Микрон), он превосходит перспективный ускоритель Eyeriss (техпроцесс 65нм, TSMC)

по энергоэффективности и по производительности. При этом Алтай не сильно уступает самому большому из существующих нейроморфному чипу IBM TrueNorth (техпроцесс 28нм, Samsung).

## Заключение

Нейрочипы сейчас находятся на той же стадии, на которой микропроцессоры были в 1980-х. Эту область в ближайшее время ожидает бурный рост [Markets and Markets, 2015]. Нейрочипы будут активно использоваться в системах зрения, управления и навигации робототехнических комплексов.

Проект по разработке нейроморфного чипа "Алтай" находится на стадии верификации HDL-кода, и в данный момент получены только расчетные данные по энергопотреблению и вычислительной мощности. Но уже сейчас чип "Алтай" можно рассматривать как один из кандидатов для использования во встраиваемых системах, где требуется реализовывать решение неформализуемых или плохо формализуемых задач (например, задачи распознавания в системах технического зрения) с высоким быстродействием и низким энергопотреблением. Низкое энергопотребление и небольшие габариты позволяют использовать его в системах технического зрения даже в небольших устройствах, например, в "умных" боеприпасах, малых БПЛА и т.п. Кроме систем технического зрения данный чип может использоваться в бортовых подсистемах управления и диагностики.

## Список литературы

**[Зверев 2007]** Зверев Г.Н. К обобщенной теории обработки наблюдений // Нефтепромысловая геофизика. – М.: ИГ и РГИ, 2007.

**[Хайкин, 2006]** Хайкин С. Нейронные сети: полный курс, 2-е издание, Пер. с англ. - М.: Издательский дом "Вильямс", 2006. - 1104с. ISBN 5-8459-0890-6(рус.)

**[Arthur et al., 2012]** Arthur J., Merolla P., Akopyan F., Alvarez R., Cassidy A., Chandra S., Esser S., Imam N., Risk W., Rubin D, Manohar R., Modha D. Building block of a programmable neuromorphic substrate: A digital neurosynaptic core // Neural Networks (IJCNN), The 2012 International Joint Conference on Year: 2012 Pages: 1 - 8, DOI: 10.1109/IJCNN.2012.6252637

**[Cassidy et al., 2013]** CassidyA., Merolla P., Arthur J., Esser S., Jackson B., Alvarez-Icaza R., Datta P., Sawada J., Wong T., Feldman V.,Amir A.,Rubin D, Akopyan F., McQuinn E., Risk W., Modha D. Cognitive computing building block: A versatile and efficient digital neuron model for neurosynaptic cores // Neural Networks (IJCNN), The 2013 International Joint Conference on Year: 2013 Pages: 1 - 10, DOI: 10.1109/IJCNN.2013.6707077.

**[Chen et al., 2016]** Yu-Hsin Chen, Tushar Krishna, Joel Emer, Vivienne Sze Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural

Networks // International Solid-State Circuits Conference 2016.

**[Esser et al., 2016]** Steven K. Esser, Paul A. Merolla, John V. Arthur, Andrew S. Cassidy, Rathinakumar Appuswamy, Alexander Andreopoulos, David J. Berg, Jeffrey L. McKinstry, Timothy Melano, Davis R. Barch, Carmelo di Nolfo, Pallab Datta, Arnon Amir, Brian Taba, Myron D. Flickner, Dharmendra S. Modha "Convolutional Networks for Fast, Energy-Efficient Neuromorphic Computing" // arXiv preprint arXiv:1603.08270, 2016.

**[Furber et al., 2013]** Furber S, Lester D., Plana L., Gaside J., Painkras E., Temple S., Brown A. Overview of the SpiNNaker System Architecture // Computers, IEEE Transactions on Year: 2013, Volume: 62, Issue: 12, pp 2454 - 2467, DOI: 10.1109/TC.2012.142

**[Imam et al., 2012]** Imam N., Akopyan F., Arthur J., Merolla P., Manohar R., Modha D. A Digital Neurosynaptic Core Using Event-Driven QDI Circuits // Asynchronous Circuits and Systems (ASYNC), 2012 18th IEEE International Symposium on Year: 2012 Pages: 25 - 32, DOI: 10.1109/ASYNC.2012.12.

**[Han et.al, 2016]** S. Han et.al, EIE: Efficient Inference Engine on Compressed Deep Neural Network // ISCA 2016.

**[Kelly, 2013]** Kelly J., Hamm S. Smart Machines: IBM's Watson and the Era of Cognitive Computing // Columbia University Press (September 24, 2013)

**[Krizhevskyet al., 2012]** Krizhevsky, A., Sutskever, I. and Hinton, G. E. ImageNet Classification with Deep Convolutional Neural Networks // NIPS 2012: Neural Information Processing Systems

**[LeCun et al., 2015]** Yann LeCun, Yoshua Bengio, Geoffrey Hinton Deep learning // Nature Vol 521, 2015, doi:10.1038/nature14539

**[Markets and Markets, 2015]** Neuromorphic chip market global forecast to 2022 // Markets and Markets, 2015.

**[NICE, 2015]** Summary Report from 2015 Neuro-Inspired Computational Elements (NICE) Workshop.

**[Nvidia, 2015]** GPU-Based Deep Learning Inference: A Performance and Power Analysis // Nvidia Whitepaper, November 2015.

**[Wang, 2008]** Xin Wang "Designing Globally-Asynchronous LocallY-Synchronous On-Chip Communication Networks" // Tempere University of Technology, 2008, ISBN 978-952-15-2005-1.

## Авторский указатель

***Для заметок***

**Российская ассоциация искусственного интеллекта**


# ТРЕТИЙ ВСЕРОССИЙСКИЙ НАУЧНО-ПРАКТИЧЕСКИЙ СЕМИНАР «БЕСПИЛОТНЫЕ ТРАНСПОРТНЫЕ СРЕДСТВА С ЭЛЕМЕНТАМИ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА»

22-23 сентября 2016
г. Иннополис, Республика Татарстан, Россия

## Труды семинара

Издательство «Перо»
109052, Москва, Нижегородская ул., д. 29–33, стр. 27, ком. 105
Тел.: (495) 973–72–28, 665–34–36
Подписано в печать 23.08.2016.Формат 60x90/16.
Бумага офсетная. Усл. печ. л. 11,5. Тираж 150 экз. Заказ 599.
Отпечатано в ООО «Издательство «Перо»
с оригинал-макета заказчика.