УДК 004.932.2

АРХИТЕКТУРА ПРОГРАММНО-АППАРАТНОГО КОМПЛЕКСА БЕСПИЛОТНОГО АВТОМОБИЛЯ ДЛЯ КОНКУРСА «ЗИМНИЙ ГОРОД»

А.К. Буйвал (*a.buyval@innopolis.ru*)
А.Р. Габдуллин (*a.gabdullin@innopolis.ru*)
Р.В. Федоренко (*r.fedorenko@innopolis.ru)*
М. С. Любимов (*m.liubimov@innopolis.ru)*

Университет Иннополис, Иннополис

Аннотация. В данной статье рассматривается архитектура программно-аппаратного комплекса используемого командой Университета Иннополис для конкурса "Зимний город". Также в статье приводятся основные технические решения и их обоснования для каждой из ключевых подсистем программного комплекса: локализация, планирование маршрута и управление, распознавание объектов и прогнозирование их поведения. В статье приведены результаты тестирования разработанного программного комплекса в различных ситуациях на электромобиле КИА.[1]

Ключевые слова: беспилотный автомобиль, конкурс «Зимний город», локализация, обнаружение объектов на изображении.

Введение

Технологические конкурсы за последние десятилетия показали свою эффективность для решения сложных задач, требующих привлечения большого количество интеллектуальных и финансовых ресурсов. В качестве примера можно привести конкурс DARPA Grand Challenge [DARPA, 2004], который стал толчком для развития не только беспилотных автомобилей, но и их различных компонентов. В 2018 году правительство РФ объявило конкурс «Зимний город» направленный на преодоление технологического барьера – создание беспилотного автомобиля способного

[1] Эта работа была поддержана Министерством науки и высшего образования Российской Федерации, в рамках проекта «Разработка модульной системы дистанционного и автономного управления коммерческим транспортом совместно с комплексом аэроразведки маршрута движения на базе отечественных компонентов» (Соглашение о предоставлении субсидии № 075-10-2018-011 (№14.609.21.0100), Уникальный Идентификатор RFMEFI60917X0100)

функционировать в сложных погодных условиях. Основный критерий конкурса заключается в том, что беспилотный автомобиль должен преодолеть 50 км за 3 часа в сложных дорожных условиях. В данной статье мы представляем архитектуру программного-аппаратного комплекса беспилотного автомобиля, которую мы, как команда Университета Иннополис, используем для решения задач конкурса.

1 Архитектура аппаратной части

В качестве базы для беспилотного автомобиля мы выбрали электромобиль KIA Soul 2014. Выбор данной модели был обусловлен следующими факторами:

- емкий аккумулятор электромобиля позволяет обеспечить стабильной электроэнергией вычислительный модуль;
- возможность управление рулевой колонкой путем имитации данных с датчика крутящего момента электроусилителя руля;
- возможность управление тягой и торможением путем имитации данных с педалей газа и тормоза.

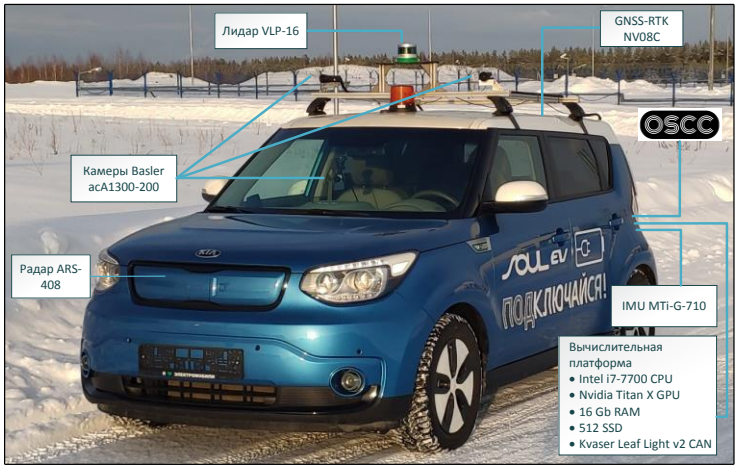

Рис. 1. Беспилотный автомобиль со схемой установленного оборудования.

Для передачи сигналов управления от вычислительного модуля к агрегатам автомобиля был использован набор модулей на базе план Arduino из открытого проекта Open Source Car Control [OSCC, 2019].

Также на автомобиль были установлены следующие сенсоры:

- 16 лучевой лазерный сканер (лидар) Velodyne VLP-16;
- 3 видеокамеры Basler AC A1300-200;

- датчик глобального позиционирования NV08C-RTK;
- датчик инерциальной системы навигации, включающий гироскоп, акселерометр и компас Xsense MTi-G-710;
- радар Continental ARS-408.

На рис. 1 представлена фотография беспилотного автомобиля с установленными датчиками.

В качестве бортового вычислительного модуля мы использовали персональный компьютер на базе процессора Intel Core i7 и с видеоадаптером Nvidia Titan X с 12 ГБ графической памяти, что позволило использовать сверточные нейронные сети для быстрой детекции объектов.

2 Архитектура программной части

В качестве базы разрабатываемой системы управления беспилотным автомобилем мы использовали открытый проект «Apollo». Данный проект использует модифицированную версию операционной системы роботов ROS, которая позволяет передавать сообщения между модулями через оперативную память, что критично для сообщений с большим количество данных. Во многом общая архитектура система представленная на рис. 2, заимствована из проекта Apollo и типична для беспилотных автомобилей. Apollo взят в качестве основы по причине наличия развитого интерфейса пользователя на основе веб-технологий, необходимых библиотек и улучшенных средств межпроцессного взаимодействия на основе protobuf, а также совместимости с симулятором беспилотных автомобилей LGSVL Simulator.

Рис. 2. Общая архитектура программной части.

2.1 Модуль локализации

Источниками данных для работы системы локализации являются:

- система глобальной спутниковой навигации GPS/ГЛОНАСС;
- кинематика реального времени (RTK);
- колесная одометрия;

– датчик инерциальной навигации;
– сенсорная система автомобиля (лазерный сканер и камеры).

Каждое из этих устройств имеет собственные условия, в которых они эффективны, но ни одно из них по отдельности, как правило, недостаточно для управления автомобилем. Так, система спутниковой навигации имеет погрешность порядка 6 метров и не работает в туннелях и в условиях плотной застройки, система RTK позволяет улучшить точность работы системы GPS да нескольких сантиметров, но вносит дополнительные ограничения, связанные с получением поправок по радиоканалу. Частота данных (до 4 Гц), получаемых от системы GPS, также недостаточна для задач управления автомобилем. При этом система инерциальной навигации и колесная одометрия могут предоставлять данные с частотой порядка 100 Гц, однако рассчитанные с их помощью координаты содержат накапливающуюся со временем ошибку. Структура модуля локализации показана на рис. 3.

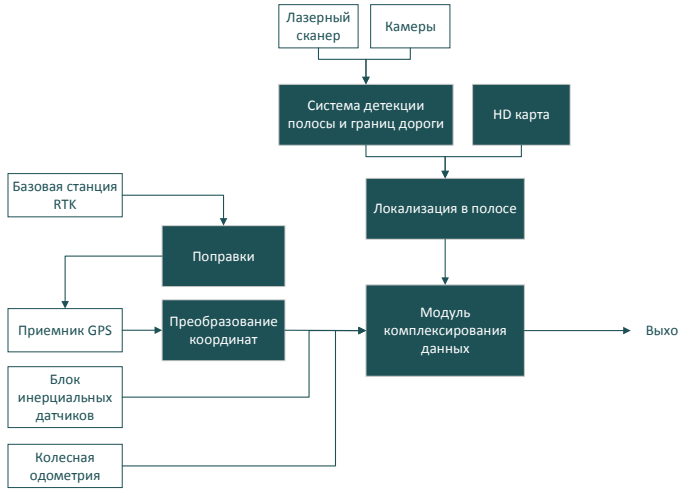

Рис. 3. Структура модуля локализации.

Таким образом, центральным модулем в системе локализации является модуль комплексирования данных различных устройств. Эта задача решается с помощью общепризнанных алгоритмов на основе фильтра Калмана [Wan et al., 2017].

Особенностью нашей реализации системы локализации автомобиля является возможность работы без системы RTK за счет использования HD-карты и системы детекции полосы и границ дороги для определения

ориентации и бокового смещения автомобиля относительно полосы на базе данных сенсорной системы (лазерного сканера и камер) без необходимости предварительного создания отдельных карт для системы локализации.

2.2 Подсистема обнаружения объектов

Общая структура модуля обнаружения объектов представлена на рис. 4.

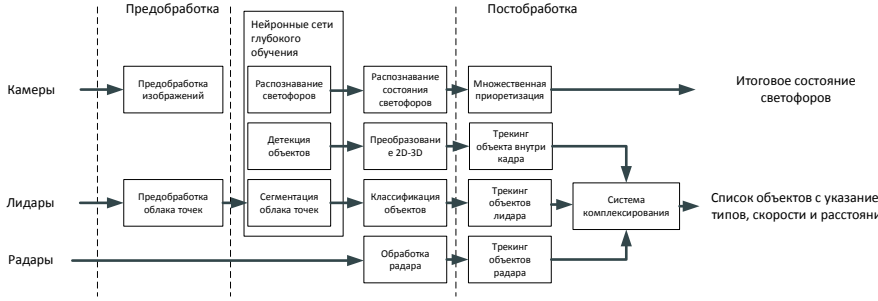

Рис. 4. Структура подсистемы обнаружения объектов.

Источниками сенсорной информации для модуля обнаружения объектов являются видеокамеры, лазерные сканеры (лидары) и радары. В состав модуля входят модуль обнаружения объектов в облаке точек, модуль обнаружения объектов по изображению с камеры, которые описаны далее, модуль комплексирования данных на основе фильтра Калмана, а также модуль распознавания состояния светофора.

2.2.1 Модуль обнаружения объектов в облаке точек

Существуют различные подходы для обнаружения объектов в облаке точек, генерируемых лидарами. Например, в работе [Li, 2016] предлагается использовать 3D сверточную нейронную сеть (СНС) для обнаружения объектов. Такой подход позволяет подавать на вход детектора облако точек практически в неизменном виде, однако такая СНС имеет большую вычислительную сложность и сильно зависит от модели лидара. В своей предыдущей работе [Buyval, 2018] мы использовали алгоритмы кластеризации на основе эвклидова расстояния для того, чтобы обнаружить кластеры и затем классифицировали их с помощью деревья решений. Однако такой подход плохо показал себя при большом количестве близко расположенных объектов.

В данной работе мы решили использовать СНС, но заранее трансформировать 3D облако точек в 2D поле признаков. Данное поле признаков представляет собой сетку ячеек в локальных координатах X-Y с

определенным шагом. Каждая ячейка имеет следующий набор признаков (каналов):

- максимальная и средняя высота точек в ячейки;
- количество точек в ячейки;
- средняя интенсивность отражения в ячейки;
- угол и расстояние от центра СК до ячейки;
- бинарное значение описывающие свободна или занята ячейка.

Далее мы подаем полученные значение на вход полносвязной СНС, которая выполняет задачу сегментации на данном поле признаков. На выходе мы получаем значения, характеризующие наличие объекта, его класс и его характеристики в каждой ячейке.

2.2.2 Модуль обнаружения объектов по изображению с камеры

В последние годы наиболее эффективными методами обнаружения объектов являются методы, основанные на использования глубинных сверточных нейронных сетей. В целом ряде исследований и задач показана высокая эффективность данных методов, однако узким местом для их использования остается высокая требовательность к вычислительным ресурсам. В нашем исследовании использовали сверточную нейронную сеть, основанную на архитектуре YOLOv3 [Redmon, 2018], т.к. она обеспечивает наилучшее соотношение точность/скорость среди существующих архитектур.

Помимо определения класса объекта и его положения на изображении важно также определить геометрические размеры объекта, а также его ориентацию на изображении. Зная эти характеристики объекта, мы затем сможем восстановить положение ограничивающего параллелепипеда в системе координат карты и передать эти данные системы трекинга.

Для того, чтобы определять геометрические размеры объекта мы добавили 3 дополнительные размерности в каждой из выходных сверточных слоев. Каждая дополнительная размерность отвечает за регрессию отклонения каждого геометрического размера от среднего размера определенного для каждого класса.

Для определения ориентации объекта мы добавили в выходные слои дополнительные 8 классов, представляющих грубую оценку ориентации, а также дополнительную размерность для коррекции угла каждого класса.

В работе [Mousavian, 2017] было акцентировано внимание на то, что визуально ориентация машина зависит не только от физического расположения автомобиля в пространстве, но также относительного положения машины от центра изображения. Для решения этой проблемы мы определяем видимую ориентацию объекта с помощью СНС, а затем, чтобы получить реальную ориентацию объекта добавляем коррекцию, основанную на положении объекта в кадре относительно центра кадра.

Таким образом окончательная ориентация объекта рассчитывается по следующей формуле:

$$\theta = \theta_{cl} + \theta_{corr} + \theta_{ray}$$

где θ_{cl} – ориентация соответствующая классу ориентацию с максимальной уверенностью на выходе СНС, θ_{corr} – коррекция ориентации для соответствующего класса, θ_{ray} – коррекция ориентации в соответствии с относительным положением объекта на кадре.

Значение θ_{ray} определяется по следующей формуле:

$$\theta_{ray} = \text{atan}(\frac{cx}{F})$$

где cx – отклонение центра объекта от центра кадра в пикселях, F – фокусное расстояние камеры в пикселях.

Финальная структура выходного слоя нашей сверточной нейронной сети представлена на рис. 5.

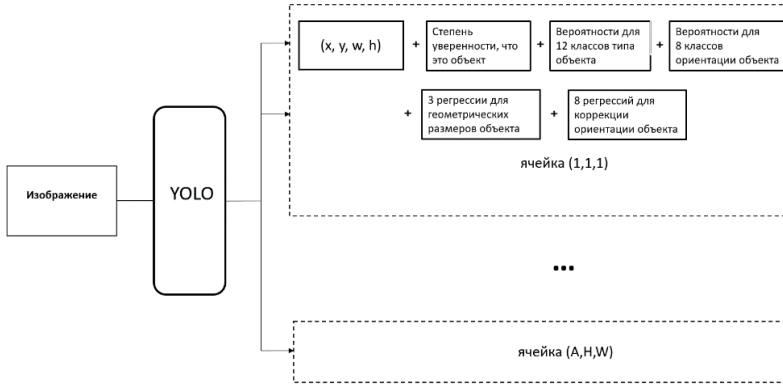

Рис. 5. Структура выходного слоя СНС.

Для обучения СНС мы использовали набор данных KITTI, как один из немногих наборов данных, который содержит данные об ориентации объекта и его геометрических размеров. Так как набор данных KITTI содержит только изображения в летний период с хорошей освещенностью, мы дополнили его собственным набором, собранным на территории города Иннополис в зимний период в дневное и ночное время.

На основе полученных с помощью СНС 2D координат, физических размеров объекта и его ориентации в пространстве далее мы делаем преобразование в 3D координаты в системе координат карты. На рис. 6 представлен пример обнаружения нескольких объектов в 3D координатах.

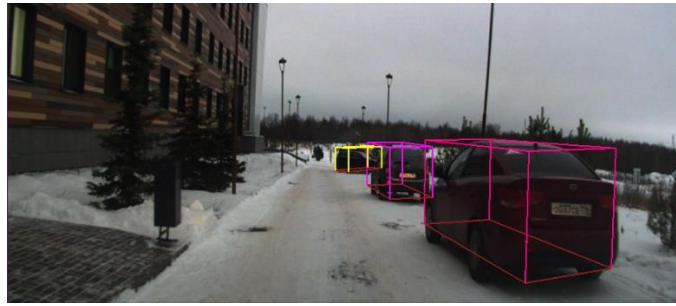

Рис. 6. Пример обнаружения объектов в 3D координатах.

2.2.3 Модуль распознавания состояния светофора

Данный модуль в качестве входных данных получает координаты обнаруженных СНС светофоров в системе координат изображения. На основе заранее известных координат светофоров в мировой системе координат (СК) мы фильтруем найденные на изображении светофоры таким образом, чтобы оставить только те, которые относятся к нашему маршруту движения. Затем изображение светофора передается на СНС, осуществляющую классификацию состояния светофора. Полученный результат затем обрабатывается с учетом известной логики переключения светофора, чтобы исключить ложные переключения.

2.3 Модули планирования траектории и управления

На момент написания данной статьи мы использовали модуль планирования из проекта Apollo. Он состоит из нескольких последовательных этапов:

- на основе базовой траектории, полученной из заданного пользователем маршрута и с учетом расположения препятствий, строится граф с минимальной стоимостью движения;
- через точки графа строится сплайн, также с минимальной стоимостью пути;
- для построенного сплайна строится профиль скорости по нему с учетом заданных ограничений на скорость и ускорения.

Полученная траектория с профилем скорости передается модулю управления. На текущий момент мы также использовали стандартный модуль управления проекта Apollo, который использует 2 независимых регулятора для управления продольной скоростью и для управления боковым отклонением от траектории. В качестве регулятора продольной скорости используется ПИД регулятор с калибровочной таблицей для

преобразования требуемого ускорения в значения нажатия педалей газа и тормоза. Регулятор бокового отклонения построен по принципу LQR регулятора и использует внутри простую динамическую модель автомобиля с 2-мя колесами.

В ходе экспериментов мы столкнулись с рядом сложностей в работе данных модулей. Наиболее серьезной проблемой было то, что модуль планирования часто выдавал абсолютно неприемлемую траекторию. В основном это было связано с тем, что система построения сплайна не могла найти оптимальное решение. Также нас не устроило то, что время планирования траектории было от 100 до 300 мс в зависимости от ситуации. В связи с этим мы решили заменить оба модуля на единый построенный по принципу Model Predictive Control. Использование подобного подхода уже показало свою эффективность в ряде наших исследований, в частности в [Buyval, 2017].

2.4 Человеко-машинный интерфейс

Для взаимодействия оператора автомобиля с системой управления используется веб-приложение (Dreamview) построенное на базе фреймворка react [React, 2019]. Оно реализовано как веб-приложение, запускаемое в браузере.

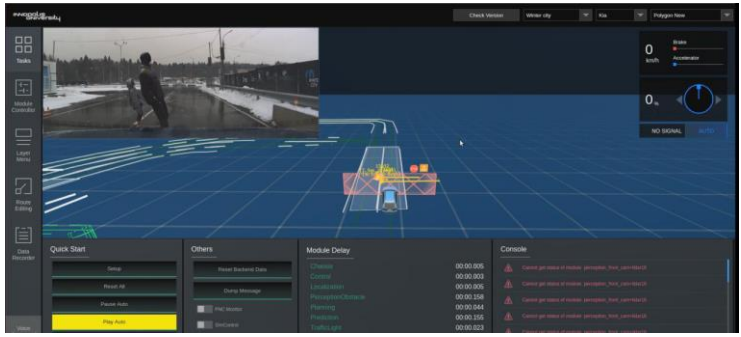

Рис. 7. Web-интерфейс программной системы.

Таким образом оно доступно для оператора как в автомобиле, так и вне его. Серверная часть данного приложения взаимодействует с системой управления автомобилем через каналы связи ROS/protobuf. Интерфейс позволяет отображать состояние запущенных программных модулей, позицию и ориентацию автомобиля на карте, препятствия и их прогнозируемые траектории в системе координат мира.

На рис. 7 показан интерфейс Dreamview в режиме автономного движения. В верхнем левом углу можно видеть изображение с курсовой

камеры. Планируемая траектория движения изображена кривой голубого цвета. Состояние распознанного курсового светофора изображено в верхнем правом углу.

3 Заключение

Конкурс «Зимний город» предоставляет возможности испытаний беспилотных автомобилей на полигоне, приближенном к городским условиям. Участие в таком конкурсе позволяет проверить на практике описанные в данной статье алгоритмы и подходы, найти их недостатки и способы улучшения.

Испытания представленной системы на полигоне показало, что автомобиль способен двигаться в автономном режиме даже в непростых погодных условиях.

Дальнейшая работа должна быть связана, прежде всего, с устранением выявленных недостатков и ограничений, связанных с улучшением надежности системы, необходимостью отказа от системы RTK для локализации, улучшением системы планирования и управления для избегания препятствий, а также улучшением качества обнаружения объектов за счет комплексирования данных различных сенсоров.

Список литературы

[Buyval, 2017] Buyval A. et al. Deriving overtaking strategy from nonlinear model predictive control for a race car //2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). – IEEE, 2017. – С. 2623-2628.

[Buyval, 2018] Buyval A. et al. Realtime Vehicle and Pedestrian Tracking for Didi Udacity Self-Driving Car Challenge //2018 IEEE International Conference on Robotics and Automation (ICRA). – IEEE, 2018. – С. 2064-2069.

[DARPA, 2004] DARPA Grand Challenge [Электронный ресурс] // wikipedia.org: Wikipedia, the free encyclopedia, информ.-справочный портал. URL: https://en.wikipedia.org/wiki/DARPA_Grand_Challenge (дата обращения: 01.04.2019).

[Li, 2016] Bo Li. 3d fully convolutional network for vehicle detection in point

[Mousavian, 2017] Mousavian A. et al. 3D bounding box estimation using deep learning and geometry // Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017. 2017. Vol. 2017–Janua. P. 5632–5640.

[OSCC, 2019] DARPA Open Source Car Control [Электронный ресурс] // github.com. URL: https://github.com/PolySync/oscc (дата обращения: 01.04.2019).

[React, 2019] https://reactjs.org/

[Redmon, 2018] Redmon J., Farhadi A. YOLOv3: An Incremental Improvement. 2018.

[Wan et al., 2017] Wan G. et al. Robust and Precise Vehicle Localization based on Multi-sensor Fusion in Diverse City Scenes. 2017.

cloud. arXiv preprint arXiv:1611.08069, 2016.