Столыпинский вестник

Научная статья

Original article

УДК 004.428

АРХИТЕКТУРА ПРОГРАММНОЙ СИСТЕМЫ БЕСПИЛОТНОГО АВТОМОБИЛЯ

ARCHITECTURE OF THE SOFTWARE SYSTEM OF THE SELF-DRIVING VEHICLE

Стахеева Алина Алексеевна, магистрант, Северный (Арктический) федеральный университет им. М. В. Ломоносова, г. Архангельск

Крайников Александр Николаевич, магистрант, Северный (Арктический) федеральный университет им. М. В. Ломоносова, г. Архангельск

Staheeva A.A. staheeva.a@edu.narfu.ru

Krajnikov A.N. krajnikov.a@edu.narfu.ru

Аннотация

Данная работа посвящена разработке архитектуры программной системы беспилотного автомобиля. Рассмотрены аналогичные программные решения, их отличительные особенности и принцип работы. Была рассмотрен инструментарий, системы моделирования, фреймворки программирования, системы датчиков и исполнительных органов. Дополнительно рассмотрена проблематика, цели и задачи концепции программной системы автоматического пилотирования и ассистирования при вождении. Была

определена По итогам работы была разработана общая структура программной системы, позволяющей автономно пилотировать автомобиль.

Annotation

This work is devoted to the development of the architecture of the software system of an unmanned vehicle. Similar software solutions, their distinctive features and principle of operation are considered. The tools, modeling systems, programming frameworks, systems of sensors and executive bodies were considered. In addition, the problems, goals and objectives of the concept of a software system for automatic piloting and driving assistance are considered. It was determined. Based on the results of the work, the general structure of the software system was developed, which allows autonomously piloting a car.

Ключевые слова: беспилотный автомобиль, ROS, Linux, ПО, системы управления автомобилем

Keywords: self-driving vehicle, ROS, Linux, software, vehicle control systems.

Введение

В современном мире уже никого не удивить автоматическими системами. Автоматические кофеварки, роботы-пылесосы и доставщики еды. Однако автоматический пилотируемый транспорт еще не так плотно вошел в нашу жизнь. Сильно замедляют распространение данной технологии три проблемы: опасность для жизни, отсутствие сформированной нормативно правовой базы и отсутствие массовых открытых систем управления. [1]

Опасность для жизни — это весомый фактор, однако это вытекающий фактор из двух следующих проблем. Отсутствие сформированной правовой – это временная проблема и уже сейчас она решается. Например, в России на данный момент уже разработаны несколько ГОСТов для решения данной проблемы, так же сейчас рассматривается изменение ПДД и дорожных знаков с целью адаптация участия беспилотных транспортных средств в дорожном движении. [2]

Самой важной и основополагающей проблемой является отсутствие базовой системы управления автомобилем. По мнению команды «Uber Engineering» отсутствие стандартов приводит к тому, что инженеры собирают свои инструменты на базе готовых технологий из различных сфер. Это приводит к тому, что итоговый продукт получается недостаточно гибким, тяжело поддерживаемым и недостаточно цельным. Что приводит к плохим результатам. [3]

Несколько компаний уже разрабатывают данные открытые системы. Например, команда «Uber Engineering» от компании «Uber» или компания «NVIDIA». [3]

В дальнейшей работе планируется рассмотреть аналоги данной системы, проанализировать наличие уже разработанных открытых систем, разработать концепцию систему, общий принцип ее работы. Основываясь на концепции и аналогах подобрать программный инструментарий и разработать архитектуру системы. В конце работы необходимо определить архитектуру системы и ее основные аспекты, необходимые для начала разработки.

Рассмотрение аналогов

На рынке систем для беспилотных автомобилей на данный момент ярко выделяются два продукта: «AVS» от компании «Uber» и «SDK NVIDIA DRIVE» от компании «NVIDIA». Оба продукта уже достаточно развиты, однако разрабатываются с несколько различными целями.

Система «AVS» предназначена для визуализации беспилотного транспорта. Она призвана стандартизировать системы визуализации. Человеку легко воспринимать данные вокруг него, например дорожные знаки или дорожную разметку. Однако машине, для того чтобы обработать большое количество данных, классифицировать эти данные и предсказывать движения участников дорожного движения приходится производить большое количество действий, а для того, чтобы принимать решения для приходится производить еще больше действий.

Реализация системы AVS состоит из двух слоев. Первый слой отвечает за сбор данных с датчиков и их обработку. Второй слой создает визуализацию это данных в виде графиков, таблиц и видео в зависимости от необходимости. Пример работы второго стоя можно наблюдать на рисунке 1.

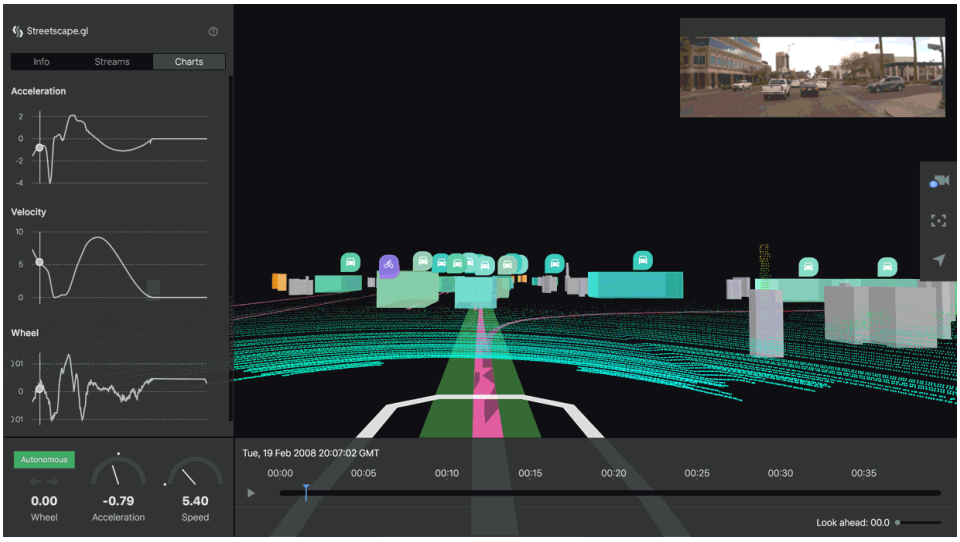

Рисунок 1 – подсистема Streetscape.gl в рамках системы AVS

Вторая система это SDK NVIDIA DRIVE. Это несколько более сложная система по сравнению с AVS. Данная система предназначена для полноценного пилотирования автомобиля и состоит из множества подсистем: операционной системы, системы сбора данных, визуализации, слежения за ситуацией в кабине, автопилотирования и ассистирования при управлении, а также системы картографической кооперации данных с множества машин. Пример работы системы на машине представлен на рисунке 2.

Рисунок 2 – Пример работы системы DRIVE

Разработка концепции

Отличительной особенностью разрабатываемой системой, по сравнению с рассмотренными будет полноценность и гибкость и простота. Рассмотренные системы либо выполняют только часть данной работы, либо узко предназначены для определенного оборудования. Разрабатываемая же система будет базировать на полном процессе от сбора данных до управления и будет состоять из простых блоков по правилам «UNIX-систем». Для универсальности системы будет использоваться стандартизированная система обмена сообщениями из взаимозаменяемых блоков, чтобы систему можно будет адаптировать для различного оборудования и целей.

В общем случае для управления автомобилем требуется четыре основных этапа: сбор данных, обработка, принятие решения, управление. Таким образом разрабатываемую систему можно условно разделить на 4 слоя. Рассмотрим подробнее каждый из них.

Первый слой – сбор данных. На данном слое необходимо реализовать систему сбора данных с датчиков в «сыром виде». При этом необходимо предусмотреть возможность подключения различных сборок датчиков. Таким

образом сбор данных должен происходить параллельно в разных потоках и при этом данные должны передаваться стандартизированном «сыром виде», чтобы не затрагивать изменения или сократить их в «обрабатывающем слое».

Второй слой – обработка данных. Этот слой служит для того, чтобы представить данные в совокупности. Этот слой не будет подстраиваться под аппаратную часть автомобиля, однако может под настраиваться. Пример работы данного слоя можно описать, как «выводы из увиденного автомобилем». Лидар установленный на крыше автомобиля может считывать расстояние до высоких удаленных объектов, деревьев, домов и так далее. Ультразвуковые датчики, установленные по периметру, могут использоваться для считывания расстояние до ближайшего, низкого объекта, дорожных конусов, соседних машин. Инфракрасные же датчики могут считывать данные до близких и удаленных объектов, но строго по прямой линии, например до впереди едущей машины. Таким образом из совокупности этих данных мы можем создать сообщение, в котором будут описываться препятствия по всему периметру и возможности движения в различных направлениях. Это сообщение будет передаваться в слой «принятия решения».

Третий слой служит для рассматривания вариантов движения, необходимости движения и возможности этого движения, а также принятия конкретного решения куда двигаться, как и двигаться ли вообще. Следует отметить, что в данном случае мы описываем систему как образ, рассматривая узко направленные случаи. Например, у автомобиля есть множество разных исполнительный органов кроме двигателя, фары, сигналы и прочее.

Четвертый слой предназначен для передачи сообщения от третьего слоя на исполнительные органы. Этот слой блоков программы весьма индивидуален для каждого исполнительного органа вследствие чего должен быть весьма универсален и/или заменяем, чтобы сохранить гибкость системы в различных аппаратных комплектациях автомобиля.

Подбор инструментов

Реализовать данную систему с нуля довольно сложная задача. вследствие чего было принято решение часть инструментов системы заменить на готовые фреймворки.

Общим связующим звеном было принято выбрать фреймворк ROS2. Данный фреймворк разрабатывается для программирования робототехники. Он реализуется систему распределенных блоков «node» обменивающихся стандартизированными сообщениями. Данный фреймворк поддерживается UNIX философию ПО, а также обладает обширным обществом пользователей, выкладывающих свои наработки в свободный доступ. Таким образом он подходит для первичной реализации системы, облегчая программирование и связь с внешними инструментами. [5]

В качестве системы управления автопилотом было принято использовать веб систему на базе фреймворка «Django». Данный фреймворк позволит создать веб-сервер, который будет отображать всю необходимую информацию в качестве веб-страниц. Это создаст возможность реализации гибкой системы управления автомобилем с множества современных устройств. [6]

В рамках беспилотных автомобилей отдельное место занимают камеры. Определение знаков, дорожной разметки, пешеходов и велосипедистов, а также автомобилей происходит именно из видео потока. Для работы с видео потоком было принято решение использовать фреймворк «OpenCV». Это один из самых популярных фреймворков компьютерного зрения имеющий большое количество методов обработки данных. [7]

Таким образом определен основный набор технологий не разрабатываемых самостоятельно, но входящих в систему в качестве зависимостей.

Общая архитектура системы

Общая архитектура уже описана в предыдущих блоках, однако требует уточнения. Как мы разобрали выше, система будет состоять из четырех слоев. Визуализация этих слоев показана на рисунке 4.

Вне слоев находятся датчики и исполнительные устройства. Это сделано в связи с разработка универсальной системы управления. Первый и четвертый слой — это промежуточные слои – в них работают программы адаптации данных для системы и для исполнительных органов. Второй слой – постобработка данных и анализ. Также в этом слое отдельно выделены некоторые стабильно участвующие системы – GPS и ГИС. В третьем слое и происходит принятие решений о действиях. Также отдельно выделены две элемента – Логи системы (история работы) и GUI (Графический интерфейс робота).

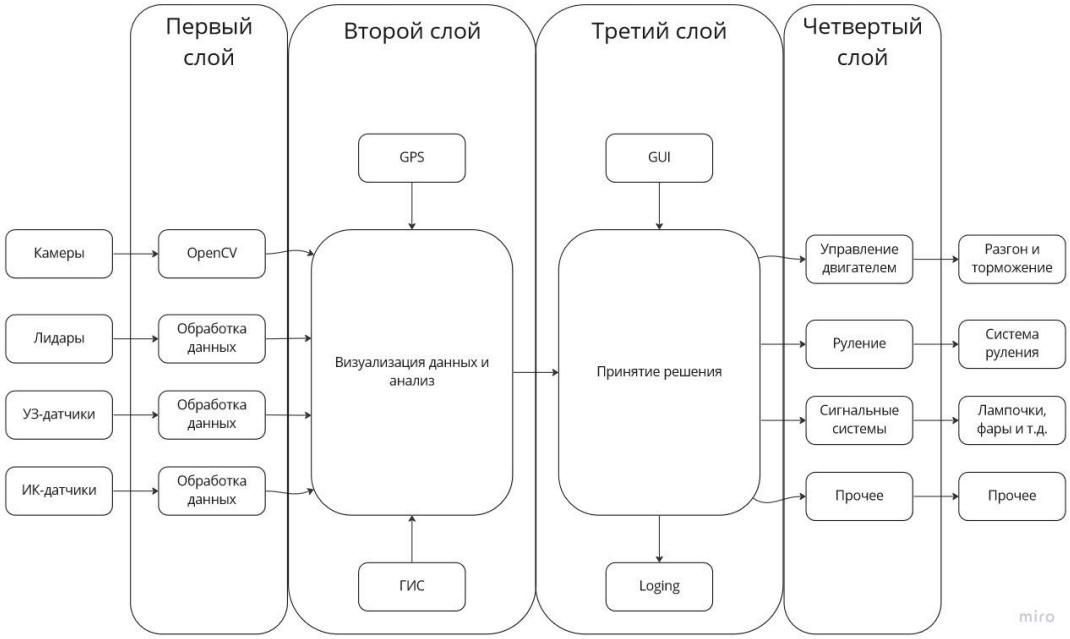

Рисунок 3 – Схема системы беспилотного автомобиля

Дополнительно введенные системы отвечают за отладку и некоторый функционал, необходимый для работы системы, они не входят в стандартную систему ввода вывода. Так или иначе эти системы не могут быть не реализованы в системе, хотя могут быть представлены в не явном виде. Например, система GPS не обязательно должна опирать на данные со спутника, для систем симуляции можно использовать внутри системную информации о положении объекта в модели города.

Заключение

В результате проделанной работы была определена обобщенная структура системы беспилотного автомобиля, определены основные зависимости и используемые сторонние инструменты, а также проанализированы ведущие представители данных систем.

Литература

1. Гальченко Г. А., Пиховкин А. С. Эпоха автономных беспилотных автомобилей: проблемы и перспективы //ИНТЕГРАЦИЯ НАУКИ, ОБЩЕСТВА, ПРОИЗВОДСТВА И ПРОМЫШЛЕННОСТИ. – 2018. – С. 17–19;
2. TADVISER [Электронный ресурс]: [офиц. сайт] / Беспилотные автомобили в России – Электрон. дан. – Режим доступа: https://www.tadviser.ru/index.php/Статья:Беспилотные_автомобили_в_России, свободный (дата обращения: 30.12.2022). – Загл. с экрана;
3. HABR [Электронный ресурс]: [офиц. сайт] / Визуализация данных для беспилотного транспорта с открытым исходным кодом от Uber – Электрон. дан. – Режим доступа: https://habr.com/ru/company/itelma/blog/497542/, свободный (дата обращения: 30.12.2022). – Загл. с экрана;
4. NVIDIA [Электронный ресурс]: [офиц. сайт] / ПО для беспилотных автомобилей – Электрон. дан. – Режим доступа: https://www.nvidia.com/ru-ru/self-driving-cars/drive-platform/software/, свободный (дата обращения: 30.12.2022). – Загл. с экрана;
5. ROS 2 Documentation [Электронный ресурс]: [офиц. сайт] / Foxy – Электрон. дан. – Режим доступа: https://docs.ros.org/en/foxy/index.html, свободный (дата обращения: 30.12.2022). – Загл. с экрана;
6. Django [Электронный ресурс]: [офиц. сайт] / Django documentation – Электрон. дан. – Режим доступа: https://docs.djangoproject.com/en/4.1/, свободный (дата обращения: 30.12.2022). – Загл. с экрана;
7. OpenCV [Электронный ресурс]: [офиц. сайт] / OpenCV-Python Tutorials – Электрон. дан. – Режим доступа: https://docs.opencv.org/4.x/d6/d00/tutorial_py_root.html, свободный (дата

обращения: 30.12.2022). – Загл. с экрана;

Literature

1. Galchenko G. A., Pikhovkin A. S. The era of autonomous unmanned vehicles: problems and prospects // INTEGRATION OF SCIENCE, SOCIETY, PRODUCTION AND INDUSTRY. - 2018. - P. 17–19;
2. TADVISER [Electronic resource]: [official. site] / Unmanned vehicles in Russia - Electronic data – Access mode: https://www.tadviser.ru/index.php/ Article: Unmanned vehicles_in_Russia, free (date of access: 12/30/2022). - From the start screen;
3. HABR [Electronic resource]: [official. site] / Data visualization for open source unmanned vehicles from Uber - Electronic data – Access mode: https://habr.com/ru/company/itelma /blog/497542/, free (date of access: 12/30/2022). - From the start screen;
4. NVIDIA [Electronic resource]: [official. site] / Software for unmanned vehicles - Electronic data – Access mode: https://www.nvidia.com/ru-ru/self-driving-cars/drive-platform/software/, free (date of access: 12/30/2022). - Title. from the screen;
5. ROS 2 Documentation [Electronic resource]: [official. site] / Foxy - Electronic data – Access mode: https://docs.ros.org/en/foxy/index.html, free (date of access: 12/30/2022). - From the start screen;
6. Django [Electronic resource]: [official. site] / Django documentation - Electronic data – Access mode: https://docs.djangoproject.com/en/4.1/, free (date of access: 12/30/2022). - From the start screen;
7. OpenCV [Electronic resource]: [official. website] / OpenCV-Python Tutorials – Electronic data – Access mode: https://docs.opencv.org/4.x/d6/d00/tutorial_py_root.html, free (accessed 12/30/2022). - From the start screen.

Для цитирования: Стахеева А.А., Крайников А. Н. АРХИТЕКТУРА ПРОГРАММНОЙ СИСТЕМЫ БЕСПИЛОТНОГО АВТОМОБИЛЯ// Научный сетевой журнал «Столыпинский вестник» №2/2023.