

Санкт-Петербургский государственный университет

Кафедра системного программирования

Группа 21.Б10-мм

Построение модели границ полосы с учетом контекста дороги

Житнухина Мария Андреевна

Отчёт по учебной практике
в форме «Решение»

Научный руководитель:
доцент кафедры системного программирования, к.т.н., Ю. В. Литвинов

Консультант:
инженер-программист, АО «Кама», М. С. Осечкина

Санкт-Петербург
2024

Оглавление

Введение	3
1. Постановка задачи	4
2. Обзор	5
2.1. Обзор моделей дороги	5
2.2. Выбранная модель полосы	7
2.3. Использование карты местности	8
3. Реализация	9
4. Апробация	15
Заключение	17
Список литературы	18

Введение

Современные автомобили все чаще оборудуются системами помощи водителю (ADAS, Advanced driver-assistance systems), которые помогают ему управлять машиной. В частности, в некоторых таких системах присутствует функция удержания в полосе, предупреждения о выходе за границы полосы, адаптивного круиз-контроля, при которой машина следует за предыдущей, автоматическая парковка, возможность получать предупреждение об опасных ситуациях и многие другие. ADAS руководствуется данными, полученными с датчиков, и через интерфейс сообщает рекомендации пользователю.

Для некоторых функций ADAS, например, системы удержания в полосе движения (англ. Lane Control Assist), необходима информация о расположении дорожной полосы в реальном времени. Ее можно распознать по данным с видеокамеры. Разметка должна храниться в виде, который удобно обрабатывать и отслеживать. Мало просто распознать разметку на кадре: отслеживать набор точек вычислительно сложнее, чем кривую заданного заранее типа. По этой причине возникает необходимость представлять дорогу в рамках некоторой модели.

Не любая модель представления дорожной полосы представляет полную информацию о полосе. Например, представление границ полосы как двух параллельных прямых не учитывает возможность поворота и сужения полосы. Возникла необходимость построить удобную для отслеживания модель границ с учетом контекста дороги. Планируется создать прототип модели полосы, который будет использоваться компанией АО «Кама».

1. Постановка задачи

Целью данной работы является создание алгоритма построения и отслеживания полосы с учетом контекста дороги.

Для выполнения работы были поставлены следующие задачи.

- Произвести обзор существующих моделей полосы.
- Выбрать модель полосы.
- Реализовать отслеживание полосы в рамках данной модели.
- Произвести оценку качества и применимости в разных условиях.

2. Обзор

В системах помощи водителю существует ряд функций, которым необходима информация о расположении разметки относительно транспортного средства. К таким функциям могут относиться удержание в полосе, предупреждение о выходе за границы полосы, автоматическая смена полосы.

Распознанная полоса должна каким-то образом храниться в памяти бортового компьютера. Для дальнейшего удобства построения траектории и минимизации затрат памяти можно представлять полосу в рамках некоторой модели. Модель — представление полосы, удовлетворяющее ряду ограничений. Они могут касаться ее формы, например, она может представлять собой график какой-либо функции.

Поскольку полоса будет использоваться для построения траектории, то она должна удовлетворять ряду свойств.

1. Из полосы должно быть возможно получить траекторию параллельным переносом.
2. Все вычисления для построения и вычисления траектории должны выполняться в режиме реального времени.
3. Полученная траектория должна минимизировать «дергания руля». В оптимальном управлении руль поворачивается с постоянной скоростью или преимущественно не вращается.

2.1. Обзор моделей дороги

Далее будут рассмотрены существующие модели, используемые для выделения разметки в режиме реального времени. Они отличаются друг от друга вычислительной сложностью, точностью описания дороги и гибкостью.

2.1.1. Множество точек

С множеством точек сложно работать: дольше и менее точно вычисляется кривизна полосы, необходимая для построения дороги, а также оно занимает много памяти.

2.1.2. Прямые

Прямые — наиболее вычислительно простой способ описания полосы. Он точно описывает лишь прямые дороги, но не повороты.

2.1.3. Гиперболы и параболы

В статье «FPGA based Curve Lane Marking Detection for ADAS» [3] за модель дороги взята гипербола. Ее коэффициенты вычисляются посредством метода наименьших квадратов. Построение гипербол и парабол вычислительно не очень сложно и они легко сглаживаются, но эти кривые описывают не самую естественную траекторию движения для автомобиля.

2.1.4. Сплайны

В статье «Lane Detection Algorithm for Intelligent Vehicles in Complex Road Conditions and Dynamic Environments» [2] за модель дороги взят b-spline. B-spline — кусочно заданная гладкая кривая, составленная из полиномиальных кривых.

Сплайны более точно описывают дорогу, чем предыдущие модели, и могут очень точно аппроксимировать множество точек, но траектория движения по сплайну не очень естественна для автомобиля, поскольку руль будет двигаться с укорением.

2.1.5. Спирали Эйлера (клотоиды)

В статье «Fused Front Lane Trajectory Estimation Based on Current ADAS Sensor Configuration» [4] в качестве модели кривой используется спираль Эйлера, так же именуемая клотоидой. Клотоида — кривая

на плоскости, у которой кривизна изменяется прямо пропорционально длине пройденного пути. Она применяется в проектировании дорог как переходная часть перед поворотом. Это делается ради того, чтобы не было скачка центробежной силы, вызванного резким изменением радиуса кривизны. Данный вид кривой хорошо описывает повороты на дороге и удобен для вычисления кривизны, однако вычислительно сложен в построении и сглаживании.

2.1.6. Комбинация дуг и отрезков

В работе [5] в качестве модели полосы используется комбинация дуг и отрезков. Дорожная разметка разбивается на участки постоянной кривизны, а затем по ним по трем точкам строятся дуги, если кривизна ненулевая, или же отрезки по двум точкам. Однако данное решение не применимо в случае прерывистой разметки, которую часто можно встретить на дорогах.

2.1.7. Дуги или отрезки

Можно описывать полосу одной кривой — дугой или отрезком. Дуги достаточно хорошо описывают физическую модель дороги в виде свеху в случае поворота, а прямые — в случае отсутствия поворота. Данная модель с некоторыми ограничениями и была выбрана в рамках данной работы.

2.2. Выбранная модель полосы

В данной работе в качестве модели полосы были выбраны две или одна кривая одного вида: дуги или прямые с ограничением на центры или параллельность соответственно. Они были выбраны, поскольку удовлетворяют всем необходимым требованиям, а именно являются удобной траекторией для автомобиля с точки зрения затрат на управление, не занимают много памяти, и аппроксимация данными кривыми возможна в режиме реального времени. Не используется жесткая модель с концентрическими дугами или параллельными прямыми, так

как из-за погрешности нахождения разметки и погрешности определения горизонта при перепроецировании к виду сверху теряется точность и выгодно использовать менее жесткую модель. Перед фильтрацией кривые приводятся к концентрическому виду, но после нее они могут перестать быть концентрическими.

2.3. Использование карты местности

В статье «Map Based Sensor Fusion for Lane Boundary Estimation on ADAS» [1] информация о разметке берется из заранее подготовленной оффлайн-карты. Использовать такой подход для решения данной задачи не самое оптимальное решение, поскольку не существует в открытом доступе карты всей разметки в России, а ее создание очень дорогое. Локализация по ГНСС имеет слишком большую погрешность, например, в горах на серпантинах, в тоннелях точность позиционирования низка. Помимо этого, даже в обычной городской среде погрешность определения местоположения превышает ширину дороги. Из-за перечисленного ряда недостатков было решено не использовать карту местности для построения полосы, а использовать только данные с датчиков автомобиля.

3. Реализация

В качестве языка реализации был выбран язык C++, поскольку он позволяет писать достаточно производительные программы для разработки систем реального времени таких как ADAS. OpenCV — наиболее активно поддерживаемая и полная библиотека с открытым кодом для работы с изображениями на C++, и поэтому она использовалась для всей работы с изображениями. Среди аналогов можно выделить библиотеку SimpleCV, построенную на основе OpenCV, но она не обновлялась с 2014 года. Существуют Microsoft Computer Vision API, Amazon Rekognition, Google Cloud Vision API. Но перечисленные библиотеки являются платными и предназначены для распознавания объектов.

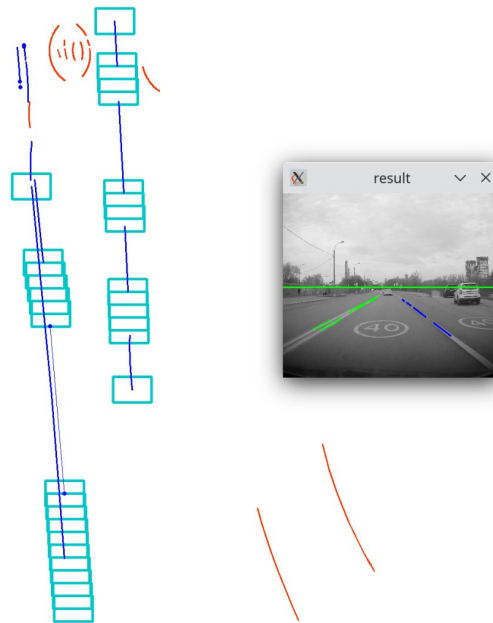


Рис. 1: Красные кривые отфильтрованы (вид сверху)

Входными данными является множество кандидатов на разметку в виде сверху (векторов точек, упорядоченных по удалению от машины). Из-за того, что данные получаем с камеры, они подвержены сферическим искажениям (дисторсии) и прямые линии в реальности не явля-

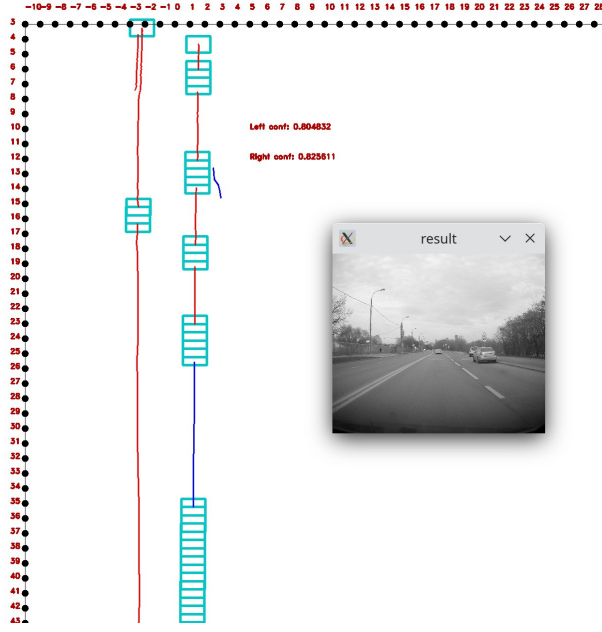


Рис. 2: Красные кривые обнаружены на предыдущих кадрах (вид сверху)

ются прямыми на изображении. Дистория происходит из-за того, что линза не плоская и может быть под углом к матрице. Убрать искажения со всего изображения невозможно в режиме реального времени, так как это займет слишком много времени. В связи с этим дисторсия убиралась лишь с распознанных фрагментов разметки, а затем накладывались обратно после перепроецирования из вида сверху.

Формулы сферических искажений выглядят следующим образом ¹:

$$x_{distorted} = x(1 + k_1r^2 + k_2r^4 + k_3r^6) \quad (1)$$

$$y_{distorted} = y(1 + k_1r^2 + k_2r^4 + k_3r^6) \quad (2)$$

$x_{distorted}$ — положение точки по оси x после искажений, а k_1, k_2, k_3 — коэффициенты дисторсии, известные после калибровки камеры. r — расстояние от оптического центра камеры до точки (x, y) в пикселях.

Встречается существенное количество ложно-положительных кандидатов на разметку, к примеру, блики на дороге. Помимо них на дороге встречаются соседние полосы и прочая разметка, которая нам не

¹https://docs.opencv.org/4.x/dc/dbb/tutorial_py_calibration.html (дата обращения: 01.06.2024)

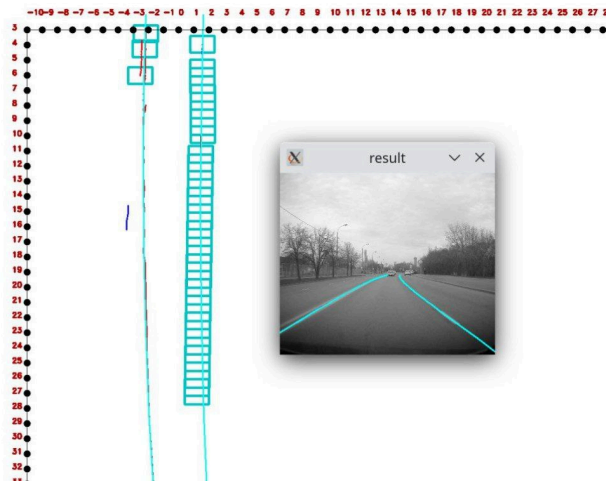


Рис. 3: Аппроксимация дугами

интересна. Было принято решение сперва провести первичную фильтрацию: были отфильтрованы кривые по кривизне, положению по оси x и по углу наклона. На Рис. 1 отфильтрованные кривые красного цвета, и видно, как были выделены части полосы слишком далеко и фрагменты круглого знака. Поскольку радиус поворота дороги не может быть меньше 15 метров по законодательству, то отфильтровывая кривые с радиусом кривизны менее 15 метров мы избавляемся от части разметки, дублирующей дорожные знаки. Ширина полосы дороги ограничена сверху ² 4.5 метрами, и поэтому полоса не может быть правее 5 метров или левее 5 от машины. Так же считаем, что машина не едет поперек полосы, поскольку выход за пределы полосы был бы определен заранее.

После первичной фильтрации все еще могут оставаться ложноположительный кандидаты. Для их определения производится отслеживание кандидатов на разметку от кадра к кадру, в качестве метрики для сравнения используется maximal deviation, то есть наибольший модуль разности по оси y . Чтобы не терять фрагменты разметки из-за сдвига автомобиля относительно дороги, используются данные одометрии. Вычисляется вектор сдвига автомобиля в метрах и изменение угла поворота между соседними кадрами по данным с датчиков и делается движение изображения. На Рис. 2 красным окрашены кандидаты

²<https://pddmaster.ru/interest/razmer-polosi.html> (дата обращения: 01.06.2024)

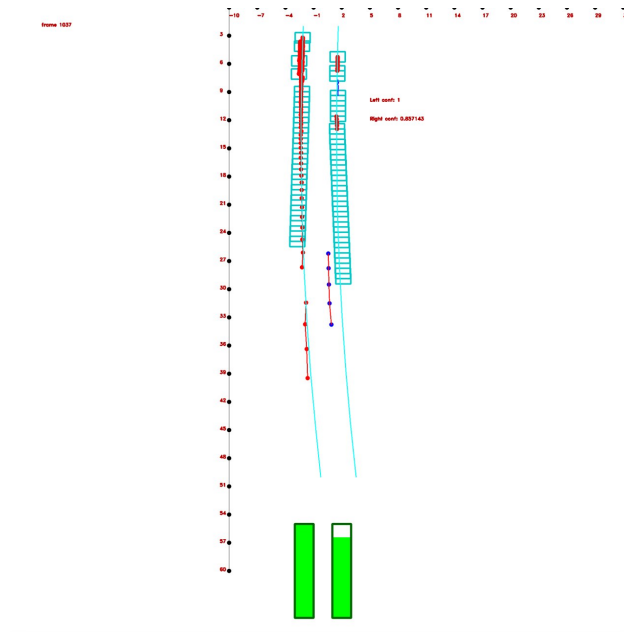


Рис. 4: Confidence модели

на разметку, которые отслеживаются от кадра к кадру, а остальные оставлены синими.

После фильтрации возникает необходимость разделить кандидатов на разметку на части левой и правой полосы. Для этого выделяются наиболее вероятные начала левой и правой полосы. За начала принимаются два кандидата на разметку, находящиеся на расстоянии не меньше двух метров и находящиеся наиболее близко к камере по оси y . Дорога точно шире двух метров, и этого ограничения снизу хватает для того, чтобы не принимать двойную разметку за разные полосы — на самом деле можно даже увеличить оценку до 3,75 м (минимальная ширина полосы). Эти кривые наиболее вероятно являются началами нужной полосы из-за ограничения угла обзора: видимые точки соседних полос находятся дальше от машины.

Далее мы предполагаем, что соседние фрагменты разметки находятся примерно на одной прямой. С помощью последовательностей окошек поиска в направлении последнего найденного фрагмента находим точку, принадлежащую следующему кандидату на разметку, и начинаем движение начиная от его конца в направлении от начала. Таким образом мы получаем две группы точек, которые делим на левые и правые.

После было решено использовать информацию, извлеченную из трекинга. Не учитываем при построении модели кандидатов на разметку, которые появились только на последнем кадре и используем тех, которые стабильно появляются и ранее участвовали в построении полосы. Далее каждое из множеств точек аппроксимируем (см. Рис. 3) окружностями с помощью алгоритма RANSAC³. Это итеративный метод, в котором выбираются случайно три точки и по ним строится окружность. Выходом алгоритма будет окружность с наибольшим числом точек на ней. Используется именно этот метод, поскольку с его помощью можно построить кривую, на которой гарантированно лежит точка из выбранного диапазона. Важно, чтобы начала кривых не смещались сильно от кадра к кадру.

Если полученный в результате аппроксимации радиус окружности превышает 2000 метров, то считаем, что на данном кадре нет поворота и дорога прямая. Аппроксимация дороги прямой осуществляется с помощью метода наименьших квадратов. Смена типа кривых в модели происходит по гистерезису.

Если были найдены точки и левой и правой части полосы в достаточном количестве для построения, приводим прямые в виде сверху к параллельному виду, а дуги к концентрическому виду.

Чтобы не было дергания полосы между кадрами, кривые сглаживаются. Для сглаживания кривых от кадра к кадру используется фильтр. Фильтруем экспоненциальным фильтром точки, принадлежащие кривым на расстоянии 2.5 м, 13 м, 20 м, и после фильтрации повторно строим по ним дуги или прямые. Были выбрано расстояние 2.5 м по причине того, что это начало видимости камеры по оси y , а остальные два были выбраны на глаз. Помимо этого, фильтруется ширина полосы. Отправляем данные в фильтр после приведения кривых к концентрическому/параллельному виду.

Для отключения отображения полосы в случае низкой уверенности в ней вычисляется confidence модели. На Рис. 4 на нижней части изображения расположены две шкалы, соответствующие уверенностям в ле-

³<https://www.mathworks.com/discovery/ransac.html> (дата обращения: 01.06.2024)

вой и правой полосе. Используется информация о количестве кандидатов, используемых в построении, которые появлялись на предыдущих кадрах хотя бы три раза, а так же оценивается, как хорошо модель ложится на кандидатов на разметку. Вычисляем процент кандидатов на разметку, которые уже были на предыдущем кадре и были включены при этом в разметку раньше — они с низкой вероятностью являются шумом. Может понадобиться отключить отображение, если полос на данных момент нет на кадре или алгоритм выдает неправдоподобный результат по какой-либо причине.

4. Апробация

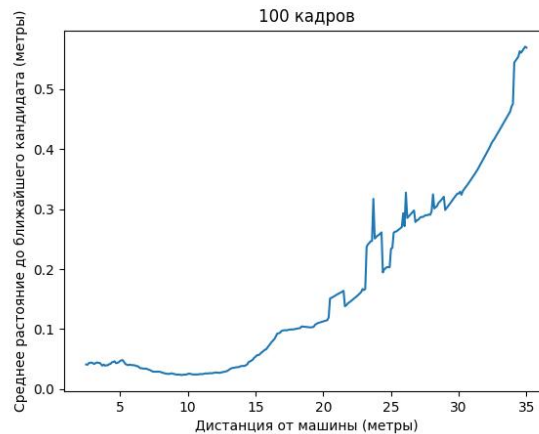


Рис. 5: Среднее расстояние до ближайшего кандидата на разметку на 100 кадрах

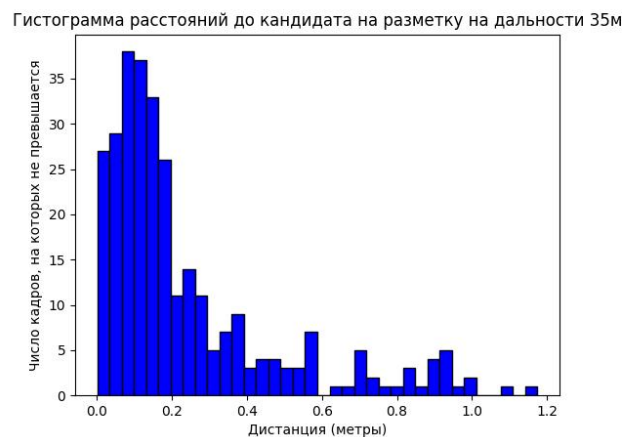


Рис. 6: Среднее расстояние до ближайшего кандидата на разметку на 100 кадрах

Для определения качества алгоритма строится гистограмма расстояний до ближайшего кандидата на разметку на разных расстояниях (см. Рис. 5) от автомобиля. Оси x на графике соответствует отдаление точки от автомобиля в метрах.

Помимо этого, строится гистограмма расстояний на зафиксированном расстоянии 35 метров от автомобиля. На расстоянии 35 метров

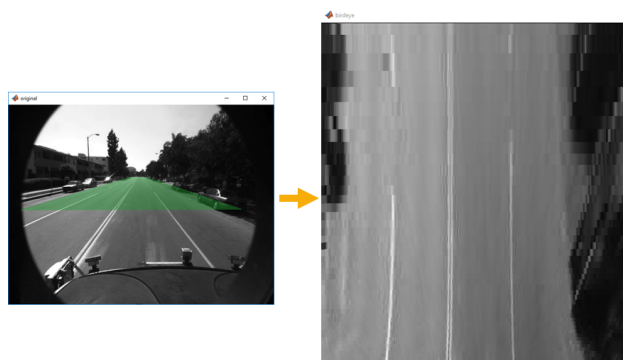


Рис. 7: Дорога после трансформации birdview

отклонение от кандидатов на разметку больше 50 см встречается значительно реже, чем меньшие отклонения, как видно на рис. 6.

Первые 30 метров алгоритм точно строит модель: средняя ошибка не превышает 0.4 метра, тогда как ширина дорожной разметки составляет от 20 до 40 сантиметров, а дальше с увеличением расстояния точность падает. Увеличение погрешности вызвано погрешностью перепроецирования на больших расстояниях от машины и сложностью выделения разметки на «размытой картинке» вдали (как на Рис. 7).

Заключение

Были выполнены следующие задачи.

- Произвести обзор существующих моделей полосы.
- Выбрать модель полосы.
- Реализовать отслеживание полосы в рамках данной модели.
- Произвести оценку качества и применимости.

В результате была выбрана модель полосы, был реализован алгоритм ее выделения, сглаживания и вычисления confidence модели, а также была произведена оценка качества алгоритма. Помимо этого, были проведены испытания на автомобиле, в которых траектория движения строилась на основе выделенной разметки.

Ссылка на репозиторий не может быть приведена из-за соглашения о неразглашении. По ссылке⁴ можно посмотреть на пример работы алгоритма.

⁴<https://drive.google.com/file/d/1EqVkl7x8PdFoTb3L-99-5LTqhnJ8pSMc/view?usp=sharing>
(дата обращения: 14.05.2024)

Список литературы

- [1] Faghi Puya. Map Based Sensor Fusion for Lane Boundary Estimation on ADAS. — 2019. — URL: <https://kth.diva-portal.org/smash/get/diva2:1770609/FULLTEXT01.pdf> (online; accessed: 2024-04-25).
- [2] Jingwei Cao Chuanxue Song Shixin Song-Feng Xiao, Peng Silun. Lane Detection Algorithm for Intelligent Vehicles in Complex Road Conditions and Dynamic Environments. — 2019. — URL: <https://www.mdpi.com/1424-8220/19/14/3166> (online; accessed: 2024-04-25).
- [3] Xin-yu WANG Jian LI Xiang-jing AN Han-gen HE. FPGA based Curve Lane Marking Detection for ADAS. — 2017. — URL: https://www.worldscientific.com/doi/10.1142/9789813206823_0032 (online; accessed: 2024-04-25).
- [4] Yuchen Liu Hao Cheng Zhiqiang Li. Fused Front Lane Trajectory Estimation Based on Current ADAS Sensor Configuration. — 2020. — URL: <https://ieeexplore.ieee.org/document/9338470> (online; accessed: 2024-04-25).
- [5] Семенов А. С. Построение и отслеживание модели границ дороги по видеопотоку. — 2022. — URL: https://se.math.spbu.ru/thesis_download?thesis_id=1138 (online; accessed: 2024-05-14).