электронный журнал
МОЛОДЕЖНЫЙ НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51038.

УДК 681.513.2

Алгоритм определения сплошных линий разметки дорожного полотна

Бошляков И.А., *студент*
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Специальная робототехника и мехатроника»

Коновалов К.В.,*студент*
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Специальная робототехника и мехатроника»

Научный руководитель: Рубцов В.И., к.т.н, доцент
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана
kafsm7@sm.bmstu.ru

Введение

В последнее время появилась потребность в учебных мобильных роботах для отработки студентами алгоритмов автоматического управления. Целями такого робота будут точное перемещение с использованием различных систем навигации и объезд препятствий на высокой скорости, таким образом студенты смогут научиться проектировать робототехнические системы используя навыки, полученные в институте.

Рис.1. E-RevoBrushless (#56087)

Проще и дешевле всего эту задачу решить на базе шасси для спортивной радиоуправляемой автомодели. Автомодель выполнена в масштабе 1/8(рис. 1) – обладает достаточными габаритами и дорожным просветом для перемещения по пересеченной местности. Позволяя студентам при выполнении курсовых и дипломных работ столкнуться с реальными трудностями при разработки мобильных робототехнических систем, в отличие от работ, разрабатываемых на базе LEGOMindstorms, ORION ROBOTICS, RoboKit 2 Roborobo, BIOLOID BeginnerKit. В настоящее время на рынке предоставлен широкий ассортимент шасси, подходящих по техническим характеристикам. Особый интерес представляет, система управления мобильного робота [8]. Главной особенностью разрабатываемого мобильного робота является движение с заданной траекторией по шоссейным дорогам. Поэтому, для движения по шоссейным дорогам необходимо разработать алгоритм определения сплошных линий разметки дорожного полотна.

Требования к алгоритму:

1. *Параметры входного сигнала:*
   - *1.1. размер кадра: не менее 340 x 270 пикселей;*
   - *1.2. частота кадров: не менее 25 кадров/сек.*
2. *Освещенность:*
   - *2.1. min освещенность сцены: не менее 100 люкс;*
   - *2.2. max освещенность сцены: не более 50000 люкс;*
3. *Наличие перспективы на фотографии;*
4. *Требования к дорожной разметке:*
   - *4.1. число полос: 2;*
   - *4.2. тип: сплошные;*
   - *4.3. min ширина полосы: 60 мм;*
5. *Время распознавания разметки кадра: не более 7 сек;*

Требования к дорожной разметке:

1.1. Ширина линии разметки согласно ГОСТ Р 51256-2011 − 0.1-0.15 м для разделения транспортных потоков противоположных направлений;

1.2. Ширина линии разметки согласно ГОСТ Р 51256-2011 − 0.2 м для обозначения края проезжей части;

1.3. Ширина полосы движения согласно ГОСТ Р 52398-2005− 3 м для не скоростных дорог;

1.4. Ширина полосы движения согласно ГОСТ Р 52398-2005− 3,75 м для автомагистральных и скоростных дорог;

Постановка задачи

Разработать алгоритм определения сплошных линий разметки дорожного полотна для автономного движения мобильного колесного робота по шоссейным дорогам на базе шасси E-RevoBrushless (#56087)со временем распознавания кадра не более 7 сек., и разрешением не менее 340 x 270 пикселей. Алгоритм является аналогом системы LaneDepartureWarningSystem(система предупреждения об отклонении от полосы движения) которая устанавливается на автомобили. Например, в автомобилях NissanJuke SL AWD 2013 года, 2013 Mercedes-Benz G63 AMG, различных моделях Audi, Toyota и иных марках авто.

Этапы обработки алгоритма

Обработка изображения проходит в несколько этапов. Это находит отображение в разделении программы на отдельные функции, каждая из которых получает в качестве входных данных результат работы предыдущих функций и производит собственную обработку. Алгоритм состоит из следующих этапов:

1. Предобработка изображения:
    a. Кадрирование
    b. Фильтрация
2. Выделение контуров на изображении
3. Сегментация линий на изображении
4. Алгоритм определения дорожной разметки

Предобработка изображения

Предобработка изображения состоит из последовательного выполнения двух этапов: *кадрирования и фильтрации*.

Кадрирование (обрезка изображения), необходима для удаления лишних деталей со снимка, например, можно убрать прохожего случайно попавшего на край фотографии. В данном случае для выделения белых полос на дорожном полотне, необходимо добиться удаления светлых участков с изображения – неба. Для этого со входного снимка выбирается вектор столбец из пикселей, который находиться по середине изображения.

На котором определяется линия горизонта. Поскольку линия горизонта, это перепад яркости пикселей, то граничный пиксел соответствует медианному значению столбца. Экспериментально было определено, что порог нужно брать немного меньше медианного значения, поскольку может быть облачно, плохая погода и т.д. Затем от найденного значения линии горизонта, отступается 15% от всей высоты изображения, для обеспечения запаса. Номер полученного значения в столбце выступает в роли координаты, для кадрирования изображения по высоте, также изображение обрезается по ширине на 10%. В случае неправильного определения линии горизонта, снимок обрезается на половину. В программе также предусмотрено кадрирование по заданным значениям.

Результат работы кадрирования изображения рис. 2.

Фильтрация

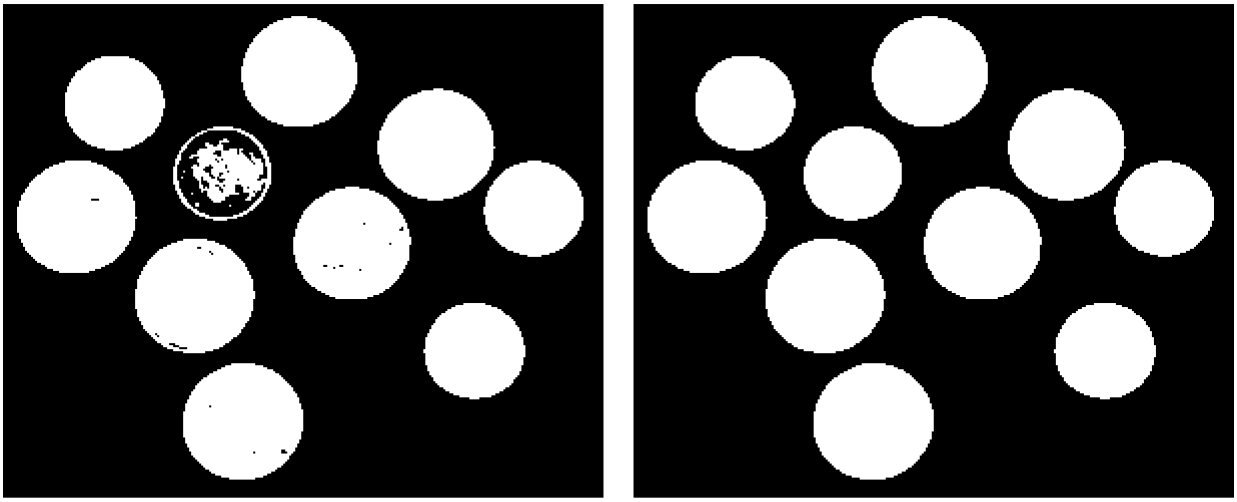
В изображениях, получаемых с видео датчика могут наводиться помехи, вызванные зернистостью самой камеры, различного рода искажения и смазы, засвеченность или наоборот слишком затемненное изображение, поэтому приходиться проводить фильтрацию изображения с целью повысить качество входного изображения, и как следствие повысить вероятность обнаружения контуров дорожного полотна. Существуют различные методы уличения качества входного изображения. В данном алгоритме, реализованы фильтры, расположенные в такой последовательности:

- морфологическая реконструкция
- стабилизация уровня яркости
- фильтрация высоких и низких частот
- повышение контрастности
- Виньеровская фильтрация

Морфологическая реконструкция имеет широкий спектр практических приложений, которые определяются выбором маркера и маски. Например, пусть в качестве маркерного изображения взято изображение f_x, которое равно 0 везде, кроме границ, а в точках границы оно равно $1 - f$:

$$f_m(x,y) = \begin{cases} 1 - f(x,y), & \text{если (x, y) лежит на границе} \\ 0 & \text{иначе} \end{cases}$$

Тогда операция $g = [R_{f^c}(f_m)]^c$имеет эффект заполнения отверстий в f, как это проиллюстрировано на рис. 3.



Original Image Filled Image

Рис.3. Пример морфологической реконструкции

Функция *imfill* выполняет эти действия автоматически при задании необязательного аргумента ’holes’ в команде[7]

g = imfill(f, ’holes’).

Эта функция действует по-разному на двоичные и полутоновые изображения, поэтому для лучшего понимания обозначений, обозначим через fB и fI, соответственно, двоичное и полутоновое изображение.

Использование синтаксиса

gB = imfill(fB, conn, ’holes’)

приводит к заполнению (т. е. она меняет значения этих пикселов на 1) дыр на двоичном изображении. *Дырой* называется множество фоновых пикселов, которых нельзя достигнуть путем заполнения фона, начиная от края изображения. *Conn* обозначает тип связности: 4 (по умолчанию) и 8. Число задействованных смежных пикселей.

Синтаксис

g = imfill(fI, conn, ’holes’)

заполняет дыры на полутоновом изображении f I. В этом случае дырой считается участок темных пикселов, окруженный более яркими пикселами. Параметр conn обозначает то же, что и раньше. Применительно к выделению белых полос на дорожном полотне, это позволяет убирать некоторые объекты фона, расположенные непосредственно на дороге, например, автомобили (рис. 4).

Поскольку сьемка может, проходить и в пасмурный день, и в яркий солнечный день, то входные изображения могут быть затемненными или соответственно пересечены,

возникает необходимость ***выравнивания уровня яркости на изображении***. Для этого исходное изображение представляется в формате данных *double*(числа с плавающей запятой) в диапазоне [0,1], в котором 0,5 – рекомендуемый уровень яркости, и соответствует 100%. Вычисляется среднее значение яркости всех пикселов, входящих в изображение, и отклонение получившегося значения в процентах от рекомендуемого. На полученную величину происходит *гамма-коррекция* изображения.

Преобразование яркости функцией intrans(пакет DIPUM)спараметром*gamma* служит для задания формы кривой (рис. 5), отображающей яркость I в яркость g [7]. Если *gamma* меньше 1, то яркость отображения смещается вверх в сторону более ярких значений а). Если *gamma* больше 1, то яркость отображения смещается вниз в сторону менее ярких значений. Если параметр *gamma* опущен, то его значение по умолчанию равно 1 (линейное отображение).

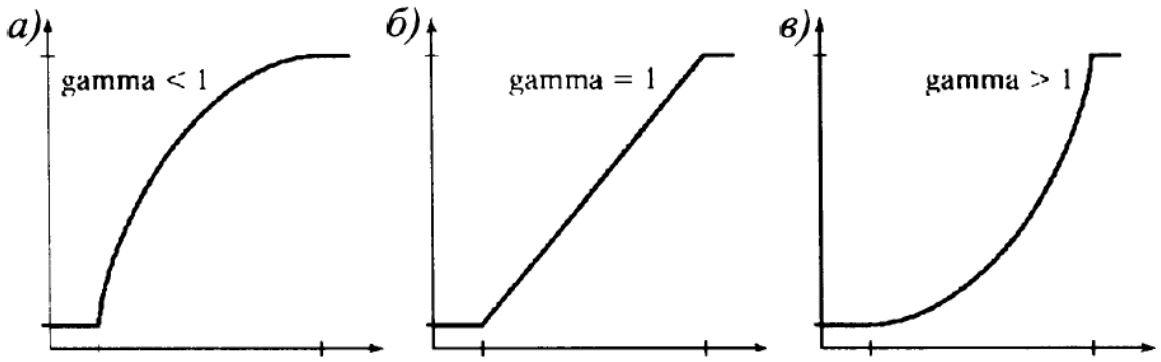


Рис. 5. Графики смещения яркости в зависимости от параметра gamma

Результат стабилизации яркости, приведен на рис. 6,7.

Фильтрация высоких и низких частот осуществляется с помощью последовательного выполнения фильтров Гаусса и Лапласа, так называемого Лапласиана-Гауссиана. Фильтр Лапласа (фильтр высоких частот) повышает резкость, а Гаусса (фильтр низких частот) ограничивает появление дополнительной зернистости на изображении.
*Оператор Лапласа*изображения $f(x,y)$ обозначается$\nabla^2 f(x,y)$ и задается формулой [7]

$$\nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2}$$

В качестве численных приближений вторых производных часто используется выражения

$$\frac{\partial^2 f}{\partial y^2} = f(x+1,y) + f(x-1,y) - 2f(x,y)$$

и

$$\frac{\partial^2 f}{\partial x^2} = f(x, y+1) + f(x, y-1) - 2f(x, y)$$

поэтому

$$\nabla^2 f = [f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1)] - 4f(x, y).$$

Данное выражение можно применить в любой точке (x,y) изображения, сделав свертку со следующей пространственной маской:

$$\begin{matrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{matrix}$$

или в общем виде:

$$\nabla^2 = \frac{4}{(\propto +1)} \begin{bmatrix} \frac{\propto}{4} & \frac{1-\propto}{4} & \frac{\propto}{4} \\ \frac{1-\propto}{4} & -1 & \frac{1-\propto}{4} \\ \frac{\propto}{4} & \frac{1-\propto}{4} & \frac{\propto}{4} \end{bmatrix}$$

Улучшение изображения с помощью оператора Лапласа производиться по формуле

$$g(x, y) = f(x, y) + c\nabla^2 f(x, y),$$

где f(x, y) — это исходное изображение, g(x, y) — улучшенное изображение, а параметр c равен 1, если центральный коэффициент маски положителен, и c = -1 в противном случае. Ниже представлен пример работы фильтра Лапласа (рис. 8). Где, а) – исходное изображение, б) – результат работы фильтра Лапласа, в) – вычитание б) из а).

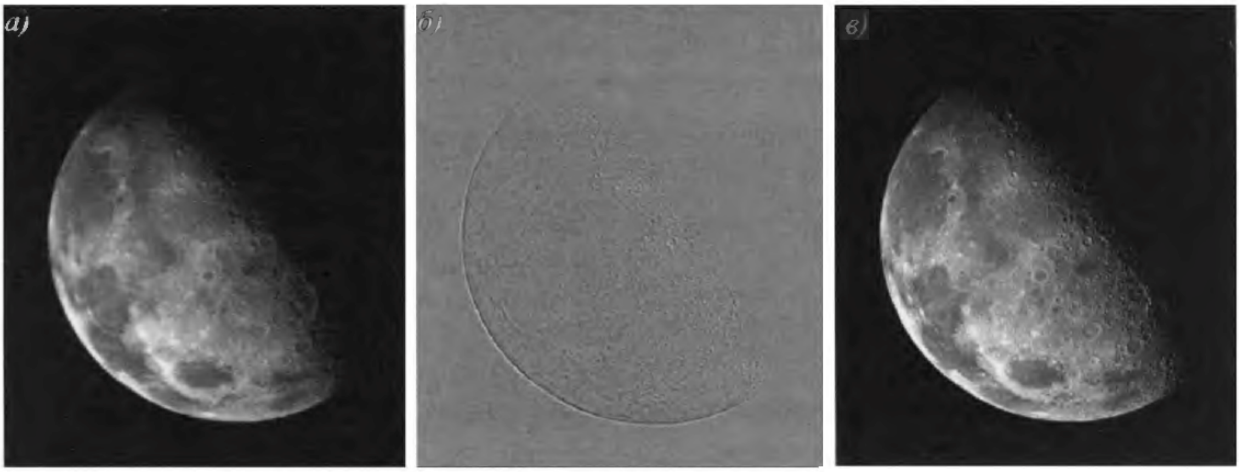


Рис. 8. Пример применения фильтра Лапласа

Фильтр Гаусса, использует одноименную функцию, которая определяется по формулам [1]:

$$h_g(n_1, n_2) = e^{\frac{-({n_1}^2+{n_2}^2)}{2\sigma^2}}$$

$$h(n_1, n_2) = \frac{h_g(n_1, n_2)}{\sum_{n_1}\sum_{n_2} h_g}$$

где h_g – значение яркости вычисляемого пиксела в скользящем окне, с координатами (n_1, n_2), а σ^2 – средне квадратичное отклонение яркости. h – значение яркости центрального пиксела в скользящем окне. При выполнении данной операции значения пикселов, несоответствующие фону, принимают среднее значение окружающих пикселов, если таких пикселов несколько, то они размываются.

Результирующий фильтр Лапласиана-Гауссиана, вычисляется по формуле [1]:

$$h(n_1, n_2) = \frac{({n_1}^2 + {n_2}^2 - 2\sigma^2)h_g(n_1, n_2)}{2\pi\sigma^6 \sum_{n_1}\sum_{n_2} h_g}$$

Результат работы фильтра Лапласиана-Гауссиана(рис. 9).

Повышение контрастности. Поскольку на полутоновом изображении линии дорожного контура являются самыми яркими элементами, то для выделения линий дорожной разметки более темные детали на изображении можно убрать. Этого можно добиться, определив значение яркости, наиболее часто встречающиеся на изображении, и все значения, расположенные в интервале от 0 до данного значения убрать. А на выходном изображении значение с 0 яркостью будет соответствовать наиболее часто встречающемуся значению яркости на изображении. Реализовать данный алгоритм просто, достаточно найти на гистограмме исходного изображения самый большой пик, и отсечь все значения яркости до него используя функцию *imadjust*. Необходимым условием работы данного алгоритма является отсутствие больших светлых участков на снимке (кадрирование изображения выполняет это условие).

Функция Id=imadjust(Is, [lowhigh], [bottomtop], gamma) создает полутоновое изображение Id путем контрастирования исходного полутонового изображения Is [3]. Значения яркости в диапазоне [lowhigh] преобразуются в значения яркости в диапазоне [bottomtop]. Значения яркости, меньшие low, принимают значение bottom, а значения яркости, большие high, принимают значение top. Значения top, bottom, low, high должны принадлежать диапазону [0,1].

Результат работы (рис. 10, 11).

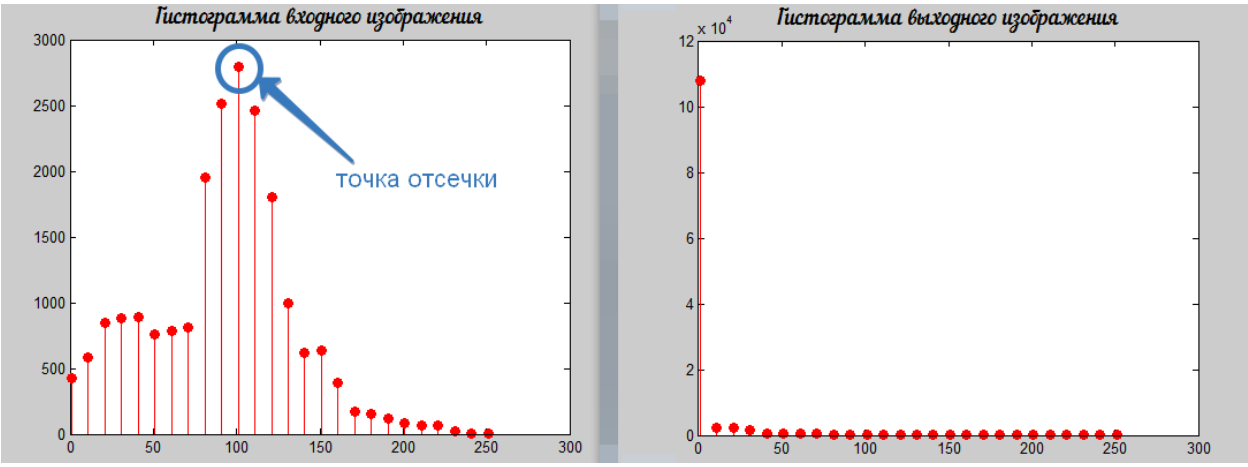


Рис. 11. Соответствующие гистограммы изображений

Винеровская фильтрация

В программе реализуется функцией *wiener2*.

Функция ***wiener2*** [2] использует алгоритм адаптивной винеровской фильтрации для подавления аддитивного гауссова белого шума. Данный алгоритм основан на статистических оценках фрагментов изображения в пределах скользящего окна размера $N \times M$ пикселей.

Для всех положений скользящего окна с центральным пикселем и координатами $n_1 \times n_2$вычисляются:

$$\mu = \frac{1}{NM} \sum_{n_1, n_2 \epsilon \eta} a(n_1, n_2)$$

$$\sigma^2 = \frac{1}{NM} \sum_{n_1, n_2 \epsilon \eta} a(n_1, n_2)^2 - \mu^2$$

$$b(n_1, n_2) = \mu + \frac{\sigma^2 - v^2}{\sigma^2} (a(n_1, n_2) - \mu),$$

где μ −средние значение яркости,σ^2 – дисперсия, $a(n_1, n_2)$ – значение яркости вычисляемого пикселя в скользящем окне, $b(n_1, n_2)$ − значение яркости центрального пикселя.

Данная формула применяется не рекурсивно для всех положений скользящего окна.

Если мощность v гауссова белого шума не задана, то она оценивается как среднее из всех σ в пределах скользящего окна.

Примечание: функции imfill и wiener2 также используются на этапе выделения контуров, но уже на бинарном изображении для увеличения толщины линий контура (рис. 12).

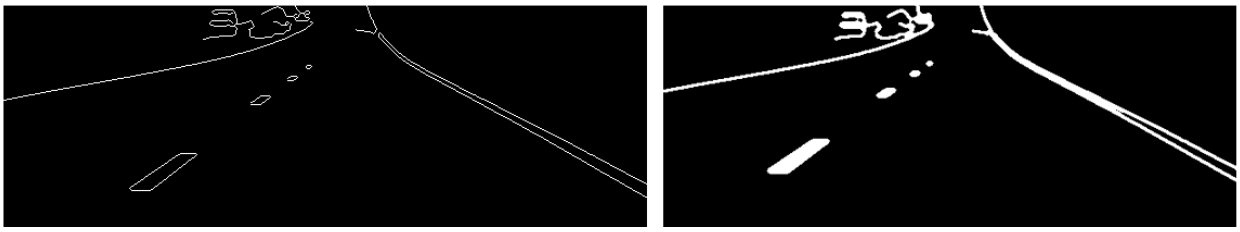


Рис. 12. Пример усиления контуров на изображении

Выделение контуров на изображении

После применения всех фильтров на этапе предобработки, на изображении выполняется сегментация линий дорожного полотна. Для этого можно провести **бинаризацию изображения** по заданному порогу. Пикселям, на изображении которые имеют значение яркости больше пороговой, будет присвоено значение 1 в формате double или 255 в uint8. Остальным пикселам будет присвоено значение 0. Таким образом происходит бинаризация изображения на выходе которой получается чёрно-белое изображение без градаций серого. Данный метод прост в реализации (его можно выполнить в среде MATLAB используя команду I=I>0.5;), не требует вычислительных ресурсов, обладает высокой скоростью выполнения. К недостаткам бинаризации можно отнести появление ошибок в различных ситуациях, например: блики на дорогах (Рис. 13. Результат применения бинаризации), яркие объекты на снимке и т.д.

В данном алгоритме был использован **метод Канни**, который осуществляет поиск границ объектов − участки изображения, в которых есть перепад яркости. Этот метод определяет контуры путём поиска локальных максимумов градиента яркости пикселов. Градиент высчитывается с использованием Перепады яркости (градиент) ищутся с помощью фильтрации по каждой из осей одномерным фильтром Лапласиан–Гауссиана. При вычислении градиента мы сталкиваемся с необходимостью выбора порогового значения градиента, который определит, какие из контуров нам отсеять, а какие принять, поскольку далеко не все границы представляют нужную нам информацию. При этом мы сталкиваемся с риском потери части, возможно, важной нам информации. Поэтому мы принимаем два значения порогового градиента: одно − для слабо выраженных границ,

другое – для ярко выраженных. Причём слабые контуры принимаются расчёт только тогда, когда они соединяются с сильными, т.е. образуют непрерывную кривую. Достоинством данного метода является возможность определения слабых границ.

Пример работы метода Канни на рис. 14.

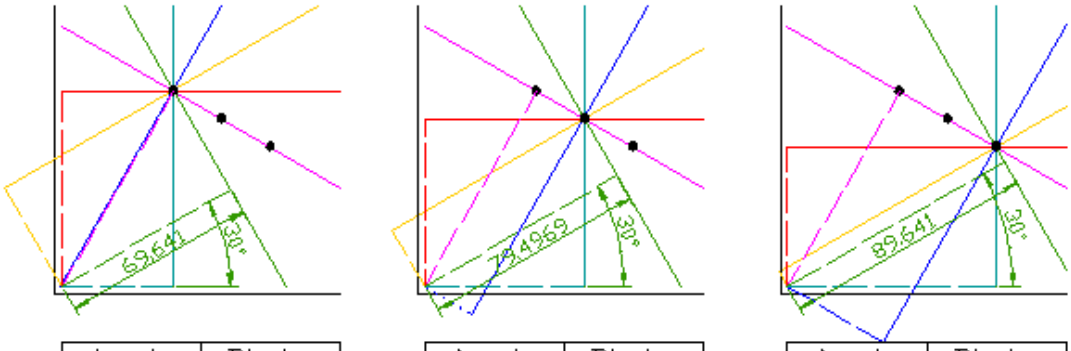
Сегментация линий на изображении

На предыдущем этапе были получены точки, находящиеся на контурах объектов, теперь надо найти линии на изображении, поскольку объектом сегментации являются линии разметки дорожного полотна и уже потом отсортировать их по определенным признакам. Для этого применяется **преобразование Хафа**[5].Данный алгоритм реализован в MATLAB с использованием функций *hough, houghpeaks, houghlines*. Алгоритм преобразования Хафа использует массив, называемый *аккумулятором (H)*, для определения присутствия прямой $y = mx + b$ на изображении BW. Данный массив возвращает функция [H, theta, rho] = hough(BW), где параметры theta (в градусах) и rho представляют массивы значений, на основе которых генерируется матрица преобразований Хафа. Размерность аккумулятора равна количеству неизвестных параметров пространства Хафа. Два измерения аккумулятора соответствуют квантизированным значениям параметров m и b. Для каждой точки и её соседей алгоритм определяет, достаточен ли вес границы в этой точке. Если да, то алгоритм вычисляет параметры прямой (для каждого thetaпо формуле$r = x\cos\theta + y\sin\theta$ вычесляется rho) и увеличивает значение в ячейке аккумулятора, соответствующей данным параметрам. Причем значением каждой ячейки является число точек плоскости X-Y проходящих через прямую с координатами (rho и theta) данной ячейки. Потом, найдя ячейки аккумулятора с максимальными значениями функцией houghpeaks в пространстве аккумулятора, могут быть определены наиболее подходящие прямые. Так как полученные прямые не содержат информацию о длине, следующим шагом является нахождение частей изображения, соответствующих найденным прямым функцией houghlines, которая из параметров rho и theta получает начальные и конечные координаты линий. Более того, из-за ошибок на этапе определения границ фигур в пространстве аккумулятора также будут содержаться ошибки. Это делает поиск подходящих линий нетривиальным.

Примечание: о дополнительных параметрах функций *hough, houghpeaks, houghlines*будет рассказано далее.

Пример работы преобразования Хафа

Рассмотрим исходное тестовое изображение из трех черных точек (рис. 15). Проверим, расположены ли точки на прямой линии.



Angle	Dist.
0	40
30	69.6
60	81.2
90	70
120	40.6
150	0.4

Angle	Dist.
0	57.1
30	79.5
60	80.5
90	60
120	23.4
150	−19.5

Angle	Dist.
0	74.6
30	89.6
60	80.6
90	50
120	6.0
150	−39.6

Рис. 15. Пример демонстрирующий принцип работы метода Хафа

- Через каждую точку проведено (для наглядности) только по шесть прямых, имеющих разный угол.
- К каждой прямой из начала координат построен перпендикуляр.
- Для всех прямых длина соответствующего перпендикуляра и его угол с осью абсцисс сведены в таблицу.
- Данные таблицы являются результатом преобразования Хафа и могут служить основой для графического представления в "пространстве Хафа".

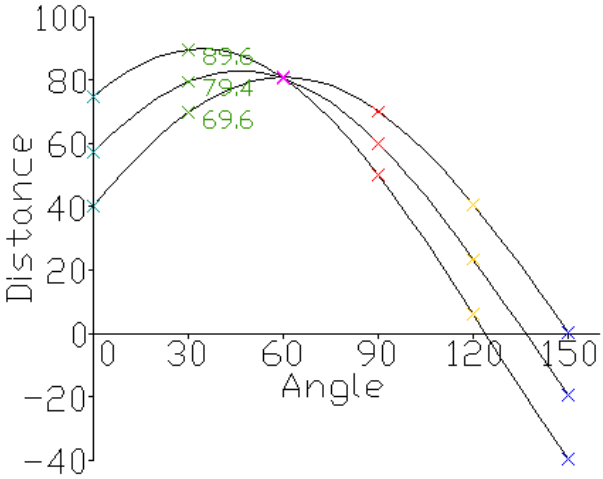


Рис. 16. Пространство Хафа

Координаты точки пересечения синусоид определяют параметры прямой, общей для проверяемых точек на исходном изображении (рис. 16).

Алгоритм определения дорожной разметки

После сегментации линий на изображении, на изображении могут быть линии, не являющиеся контурами дорожной разметки, поэтому линии надо отфильтровать по ряду признаков. Но прежде чем фильтровать полученный массив линий, определяется уровень качества снимка по количеству найденных линий. По умолчанию программа работает в режиме для *изображений с плохим качеством*, но, если на снимке функция houghpeaks нашла numpeaks линий программа переходит в режим для *изображений хорошим качеством*. При этом программа возвращается на этап выделения контуров и повторно выделяются границы методом Канни, но уже с более высоким порогом thresh. Далее на бинарном изображении границы объектов увеличиваются функциями wiener2 и imfill для лучшего обнаружения линий методом Хафа, пороги которого также подняты. Дополнительно изображение проходит через адаптивный медианный фильтр *adpmedian* для подавления помех, полученных в результате усиления контуров.

Алгоритм **фильтрации линий** схож с реализацией метода Хафа в MATLAB. Сначала создается квадратная матрица из нулевых элементов размерностью равной количеству найденных линий на изображении. После этого в 2 двух циклах (один вложен в другой) каждая линия сравнивается со всеми другими по следующим признакам:

- разница по модулю между углами пересечения линий с горизонталью не более $45°$
- имеют разную длину перпендикуляра r
- значения перпендикуляров rположительные
- углы между горизонтальной осью и перпендикуляром $-\ \theta$ имеют разные знаки
- углы пересечения линий с горизонталью лежат в диапазоне от $10.5°$ до $86°$

Если условия выполнены, то для линий с соответствующими индексами в цикле, значение разницы между углами пересечения линий с горизонталью записывается в матрицу. Далее необходимо, выбрать minэлемент > 0, для этого все 0 значения матрицы увеличиваются на 1000. Таким образом, данный параметр определяет наиболее вероятную пару линий дорожной разметки.

На рис. 17 жирными красными линиями выделена сегментированная пара линий разметки. Углы α_1и α_2 −углы пересечения линий с горизонталью, и равны соответственно

11° и 30°. Значения θ даны в градусах, а r в пикселах.

Для **определения положения транспортного средства** применяется два способа. *В первом* используется более жесткий порог разницы по модулю между углами пересечения линий с горизонталью − 35°. Если значение больше порогового, то траектория движения автомобиля пересекает линию дорожной разметки – высокая вероятность съезда с дороги. *Второй способ* определяет среднюю точку между двумя ближайшими точками расположенных на линиях дорожной разметки. На основе которой строиться вертикальная прямая – предполагаемая траектория движения. Также вычисляется точка пересечения двух линий разметки.

Для этого обе линии записываются в общем виде [6]:

$$A_1x + B_1y + C_1 = 0, \qquad A_2x + B_2y + C_2 = 0.$$

где $A_i = y_1 - y_2,\ B_i = x_2 - x_1,\ C_i = x_1y_2 - x_2y_1,$

(x_1, y_1) и (x_2, y_2)– координаты начала и конца линий известные из преобразования Хафа.

Точка пересечения в проекции на горизонтальную ось определяется по формуле

$$x = \frac{B_1C_2 - B_2C_1}{A_1B_2 - A_2B_1}$$

Затем высчитывается расстояние в пикселах между точкой пересечения и средней точкой.

Если полученное значение больше 71, то траектория автомобиля пересекает дорожную разметку (синяя линия) – высокая вероятность съезда с дороги.

Пример работы алгоритма (рис. 18, 19).

Создание графического интерфейса в среде MATLAB

Для проведения экспериментов был создан графический интерфейс. На рис. 20 показан графический интерфейс, дополнительно выводятся еще несколько окон отображающие промежуточные результаты работы, и конечный результат.

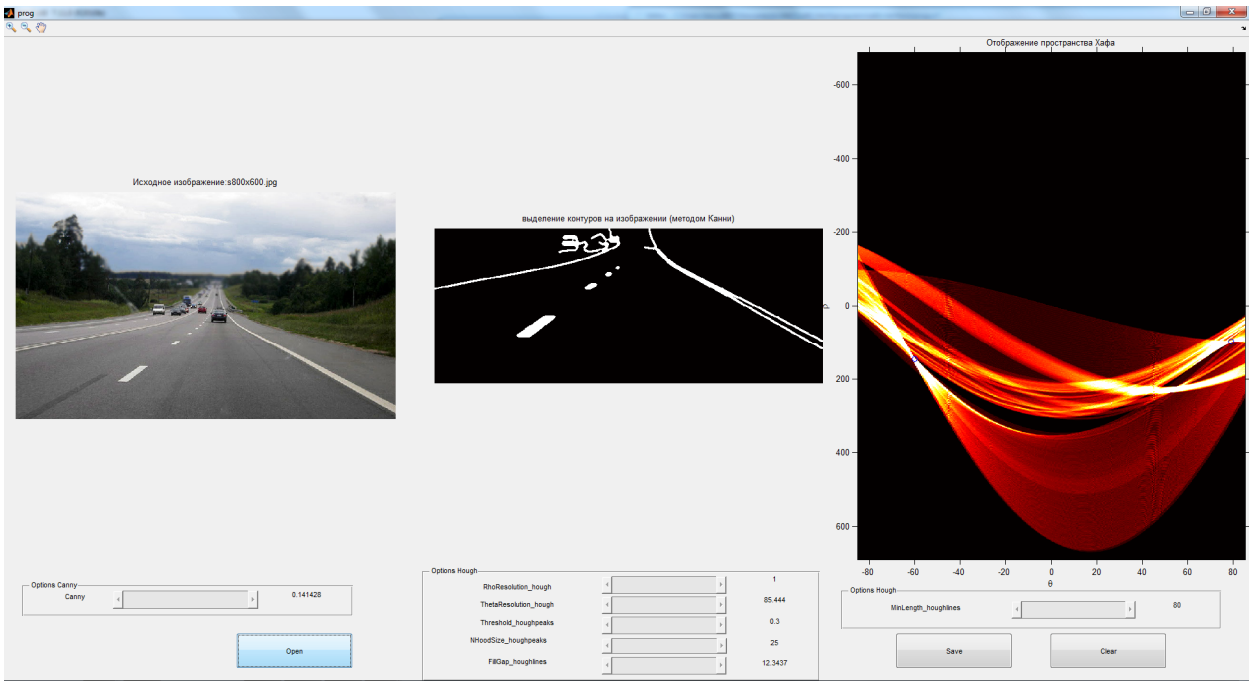


Рис. 1. Интерфейс программы

OptionsCanny устанавливает значение порога thresh (рис. 21).

Рис. 2. Scroll для регулировки параметра метода Канни

Остальные параметры регулируют параметры преобразования Хафа(рис. 22, 23).

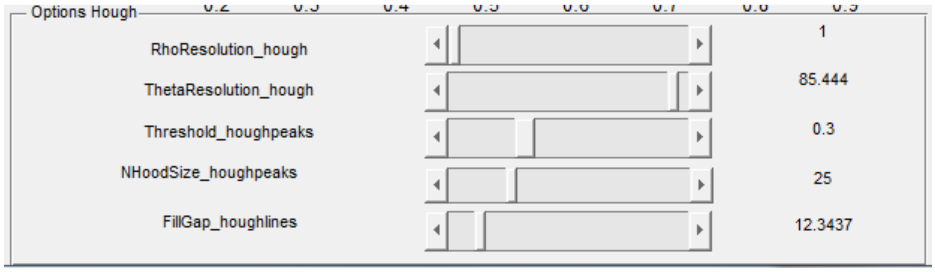


Рис. 3. Scrolls для регулировки параметров метода Хафа

Рис. 4. Scroll для регулировки параметра метода Хафа

RhoResolution_hough огранивает число возможных значений параметра rho, путем прореживания матрицы Хафа вдоль соответствующей размерности. *ThetaResolution_hough*

задает максимальный диапазон theta. В силу квантования пространства цифрового изображения и пространства параметров, а также по причине того, что края и перепады на типичных изображениях не являются совершенно прямыми, максимумы преобразования Хафа могут достигаться более чем в одной ячейке накопления. Эту сложность можно преодолеть с помощью следующей стратегии.

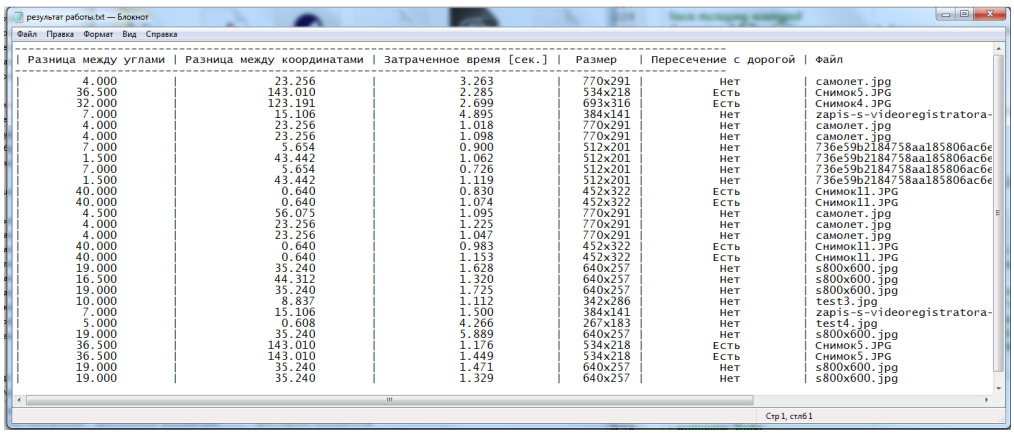
1. Найти ячейку преобразования Хафа, в которой лежит наибольшая величина, и записать ее местоположение.

2. Опорожнить (обнулить) ячейки в ближайшей окрестности положения, найденного на шаге 1.

3. Повторять шаги 1 и 2 до тех пор, пока желаемое число максимумов не будет найдено, или после достижения заданного порога.

Функция houghpeaks реализует эту стратегию. *Threshold_houghpeaks* устанавливает значение порога матрицы Хафа. *NHoodSize_houghpeaks* учитывает влияние окрестности в матрице Хафа. *FillGap_houghlines* устанавливает минимальное расстояние между линиями. *MinLength_houghlines* выставляет минимальное значение длины линии.

Все результаты можно сохранить нажатием клавиши Save. Кроме того, отчет работы программы автоматически сохраняется в текстовый файл *результаты работы* (рис. 24).



результат работы.txt — Блокнот

Файл Правка Формат Вид Справка

Разница между углами	Разница между координатами	Затраченное время [сек.]	Размер	Пересечение с дорогой	Файл
4.000	23.256	3.263	770x291	Нет	самолет.jpg
36.500	143.010	2.285	534x218	Есть	Снимок5.JPG
32.000	123.191	2.699	693x316	Есть	Снимок4.JPG
7.000	15.106	4.895	384x141	Нет	zapis-s-videoregistratora-
4.000	23.256	1.018	770x291	Нет	самолет.jpg
4.000	23.256	1.098	770x291	Нет	самолет.jpg
7.000	5.654	0.900	512x201	Нет	736e59b2184758aa185806ac6e
1.500	43.442	1.062	512x201	Нет	736e59b2184758aa185806ac6e
7.000	5.654	0.726	512x201	Нет	736e59b2184758aa185806ac6e
1.500	43.442	1.119	512x201	Нет	736e59b2184758aa185806ac6e
40.000	0.640	0.830	452x322	Есть	Снимок11.JPG
40.000	0.640	1.074	452x322	Есть	Снимок11.JPG
4.500	56.075	1.095	770x291	Нет	самолет.jpg
4.000	23.256	1.225	770x291	Нет	самолет.jpg
4.000	23.256	1.047	770x291	Нет	самолет.jpg
40.000	0.640	0.983	452x322	Есть	Снимок11.JPG
40.000	0.640	1.153	452x322	Есть	Снимок11.JPG
19.000	35.240	1.628	640x257	Нет	s800x600.jpg
16.500	44.312	1.320	640x257	Нет	s800x600.jpg
19.000	35.240	1.725	640x257	Нет	s800x600.jpg
10.000	8.837	1.112	342x286	Нет	test3.jpg
7.000	15.106	1.500	384x141	Нет	zapis-s-videoregistratora-
5.000	0.608	4.266	267x183	Нет	test4.jpg
19.000	35.240	5.889	640x257	Нет	s800x600.jpg
36.500	143.010	1.176	534x218	Есть	Снимок5.JPG
36.500	143.010	1.449	534x218	Есть	Снимок5.JPG
19.000	35.240	1.471	640x257	Нет	s800x600.jpg
19.000	35.240	1.329	640x257	Нет	s800x600.jpg

Стр 1, стлб 1

Рис. 5. Содержимое файла результат работы.txt

Выполнение серии экспериментов разработанного алгоритма

Для проведения серии экспериментов было выбрано 100 фотографий. Эксперимент учитывает следующие параметры:

- Значение разницы между углами пересечения линий дорожной разметки с горизонталью (1 способ)
- Значение расстояния между точкой пересечения линий дорожной разметки и средней линией (2 способ)
- Время, затраченное на обработку кадра
- Размер кадра (кадрированного)
- Наличие пересечения траектории движения транспортного средства линии дорожной разметки
- Ошибка 1 рода, *α errors* (на фотографии с разметкой не найдены линии дорожной разметки) [4]
- Ошибка 2 рода, *β errors* (на фотографии без разметки были найдены линии дорожной разметки) [4]
- Результат обнаружения 1 способом
- Результат обнаружения 2 способом
- Результат определения пересечения траектории движения транспортного средства линии дорожной разметки

Результаты работы сведены в таблицу:

Таблица 1

Экспериментальные данные

№	Разница между углами, [град.]	Разница между коорд., [пиксел]	Затраченное время, [сек.]	Размер кадра	Пер. с дорогой	α errors	β errors	Результат обнар. 1 способом	Результат обнар. 2 способом	Результат определения пересечения с дорогой
1	4.1	17.9	6	1492x490	Нет	−	−	+	+	+
2	5.1	54.8	5.7	1494x547	Нет	−	−	+	+	+
3	14.1	43.1	4.9	1494x470	Нет	−	−	+	+	+
4	29.9	88.2	4.1	1211x489	Есть	−	−	−	+	+
5	3.1	27.4	3.4	946x504	Нет	−	−	+	+	+
6	14.9	7.44	3.2	814x526	Нет	−	−	+	+	+
7	_	_	_	1477x845	_	+	−	_	_	_
8	36.4	50	3.9	1114x563	Есть	−	−	+	−	+
9	7.4	10.4	3.2	825x504	Нет	−	−	+	+	+
10	42.4	5.1	4.4	1028x589	Есть	−	−	+	−	+
11	19.9	132.8	3	818x498	Есть	−	−	−	+	+
12	8.9	44.3	3.2	911x473	Нет	−	−	+	+	+
13	5.4	3.7	2.3	710x350	Нет	−	−	+	+	+
14	35.4	21.7	4.1	1188x555	Есть	−	−	+	−	+
15	27.1	61.7	4.3	1238x499	Нет	−	−	−	−	-
16	1.1	3	3.5	1031x596	Нет	+	−	+	+	+
17	_	_	_	1289x917	_	+	−	_	_	_

18	1.4	12.4	3.7	1486x428	Нет	−	−	+	+	+
19	0.9	66.6	2.4	1466x418	Нет	+	−	+	+	+
20	29.1	87	3.2	1194x425	Есть	−	−	−	+	+
21	30.1	13.2	3.6	1166x500	Нет	−	−	+	+	+
22	3.1	16.5	3.5	1038x482	Нет	−	−	+	+	+
23	3.6	6.7	3.9	1046x437	Нет	−	−	+	+	+
24	29.6	32.8	3.5	1190x461	Нет	−	−	+	+	+
25	0.1	35.9	3.7	1491x446	Нет	−	−	+	+	+
26	0.1	0.7	4.2	1487x482	Нет	−	−	+	+	+
27	43.1	25.7	4.2	1239x535	Есть	+	−	+	−	+
28	_	_	_	1409x874	_	+	−	_	_	_
29	2.4	14	3.6	1097x477	Нет	−	−	+	+	+
30	1.6	12	4.1	1465x460	Нет	−	−	+	+	+
31	4.9	7.6	4.7	1536x481	Нет	−	−	+	+	+
32	3.1	38.1	4.8	1536x491	Нет	−	−	+	+	+
33	8.4	80	4.9	1526x478	Есть	−	−	+	−	-
34	0.9	1.7	4.2	1536x469	Нет	−	−	+	+	+
35	3.4	5.2	4.3	1536x491	Нет	−	−	+	+	+
36	4.9	9.9	4.3	1524x478	Нет	−	−	+	+	+
37	11.1	4.4	4.2	1536x471	Нет	+	−	+	+	+
38	6.9	13.7	3.9	1536x440	Нет	−	−	+	+	+
39	4.9	39	4	1536x448	Нет	−	−	+	+	+
40	0.1	0.2	4.3	1536x470	Нет	−	−	+	+	+
41	6.4	13.8	5.1	1536x448	Нет	−	−	+	+	+
42	13.4	31.2	4.2	1536x439	Нет	−	−	+	+	+
43	17.4	40.1	4.2	1536x437	Нет	−	−	+	+	+
44	9.9	65.8	5.1	1536x532	Нет	−	−	+	+	+
45	8.9	21.5	4	1524x424	Нет	−	−	+	+	+
46	12.1	13.1	4.4	1518x481	Нет	+	−	+	+	+
47	0.6	44.2	7.9	1526x540	Нет	−	−	+	+	+
48	8.9	16.8	4.9	1522x453	Нет	−	−	+	+	+
49	17.9	17.3	4.2	1524x406	Нет	−	−	+	+	+
50	20.4	16.8	5.1	1518X536	Нет	−	−	+	+	+
51	18.4	44.9	3.9	1536x414	Нет	−	−	+	+	+
52	8.4	142.2	6.1	1527x451	Есть	−	−	+	−	-
53	_	_	_	1909x951	_	+	−	_	_	_
54	4.1	37	4	1524x444	Нет	−	+	+	+	+
55	5.6	0.4	5.7	1521x586	Нет	−	−	+	+	+
56	2.4	13.3	4.6	1526x492	Нет	−	−	+	+	+
57	10.1	247.1	5.6	1536x495	Есть	−	−	−	+	-
58	5.6	246	4.7	1536x518	Есть	−	−	−	+	-
59	6.4	4	4.9	1536x565	Нет	−	−	+	+	+
60	3.4	6.2	4.2	1536x458	Нет	−	−	+	+	+
61	2.9	48	5.1	1536x588	Нет	−	−	+	+	+
62	9.9	0.7	5.5	1536x600	Нет	−	−	+	+	+
63	16.9	30.2	4.2	1536x463	Нет	−	−	+	+	+
64	15.4	81.4	4.3	1536x483	Есть	−	−	+	−	-
65	17.9	46.3	4.4	1536x499	Нет	−	−	+	+	+
66	0.9	62.7	4.5	1536x503	Нет	−	−	+	+	+
67	7.1	8.9	4.6	1536x501	Нет	−	−	+	+	+

68	14.4	15.6	4.5	1536x519	Нет	−	−	+	+	+
69	13.4	25.3	3.7	1536x394	Нет	−	−	+	+	+
70	6.4	33.4	5.1	1536x619	Нет	−	−	+	+	+
71	22.6	57.8	6.5	1536x462	Нет	−	−	+	+	+
72	20.1	22	5.5	1536x527	Нет	−	−	+	+	+
73	21.6	20.7	5.7	1536x579	Нет	−	−	+	+	+
74	16.1	10	5.9	1536x586	Нет	−	−	+	+	+
75	29.6	2.1	6.1	1536x596	Нет	−	−	+	+	+
76	21.1	5.5	6.1	1536x591	Нет	−	−	+	+	+
77	28.6	43.1	5.8	1536x594	Нет	−	−	+	+	+
78	25.1	19	5	1536x537	Нет	−	−	+	+	+
79	7.9	38.7	1.6	520x169	Нет	−	−	+	+	+
80	1.9	5.4	2	640x321	Нет	−	−	+	+	+
81	13.1	11.1	1.1	512x184	Нет	−	−	+	+	+
82	25.9	45.5	1	500x123	Нет	−	−	+	+	+
83	6.1	10	1.3	560x60	Нет	−	−	+	+	+
84	2.1	3.7	1.4	384x201	Нет	−	−	+	+	+
85	–	–	–	620x349	–	−	−	–	–	–
86	14.6	5.4	1.7	640x234	Нет	−	−	+	+	+
87	1.9	1.2	1.4	267x170	Нет	−	−	+	+	+
88	6.6	3.8	2	694x242	Нет	−	−	+	+	+
89	35.6	28	1.8	528x215	Есть	−	−	+	−	+
90	25.1	9.8	1.7	570x211	Нет	−	−	+	+	+
91	–	–	–	557x570	–	+	−	–	–	–
92	38.1	125.9	1.7	534x190	Есть	−	−	+	+	+
93	0.6	41	1.4	673x347	Нет	−	−	+	+	+
94	25.1	43.5	1.4	474x186	Нет	−	−	+	+	+
95	24.1	53.9	1.6	578x163	Нет	−	−	+	+	+
96	2.9	13.8	1.2	604x237	Нет	−	−	+	+	+
97	2.4	1.3	1.8	598x318	Нет	−	−	+	+	+
98	18.6	86.7	1.3	452x299	Есть	−	−	−	+	+
99	36.1	0.4	6.7	1536x622	Есть	−	−	+	−	+
100	6.6	8.2	5.1	1536x499	Нет	−	−	+	+	+

Опыт показывает, что алгоритм дал 84% верных ответов. В ходе проведения эксперимента было выявлено 10% ошибок 1 рода, и 1% 2 рода. Определение траектории движения 1 способом дало сбой в 7% случаях, 2 способом в 10%. Со средним временем распознавания 4 секунды.

Вывод: Разработанный алгоритм применим для его использования в качестве элемента системы технического зрения на мобильном колесном роботе. Возможна дальнейшая модернизация алгоритма.

Заключение

Разработан алгоритм определения сплошных линий разметки дорожного полотна. Проведено компьютерное моделирование разработанного алгоритма в среде математического моделирования Mathworks MATLAB. Выполнена серия экспериментов разработанного алгоритма.

Этапы результатов эксперимента

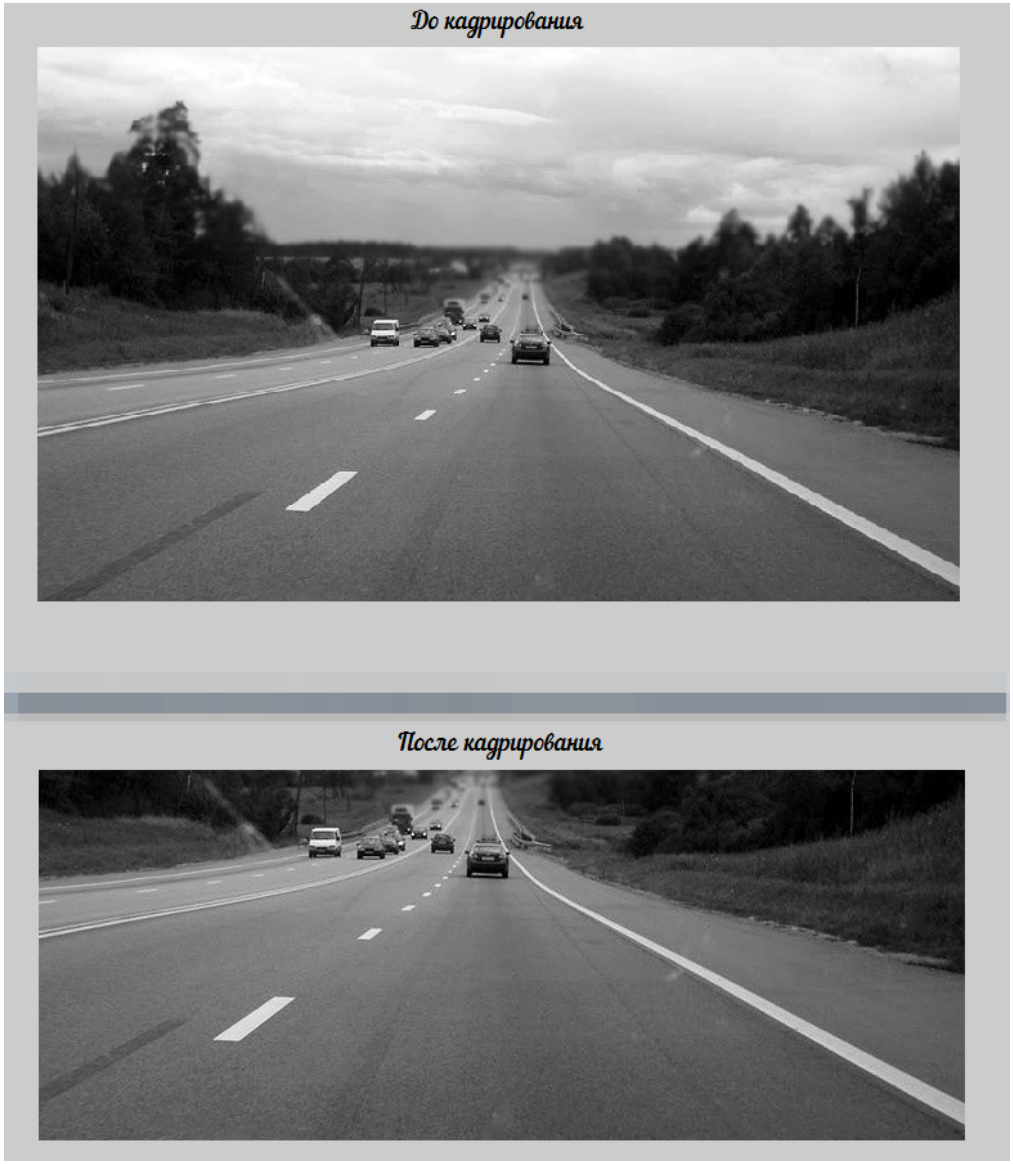


Рис. 6. Пример кадрирования

Рис. 7. Применение морфологической реконструкции к изображению с дорожным полотном

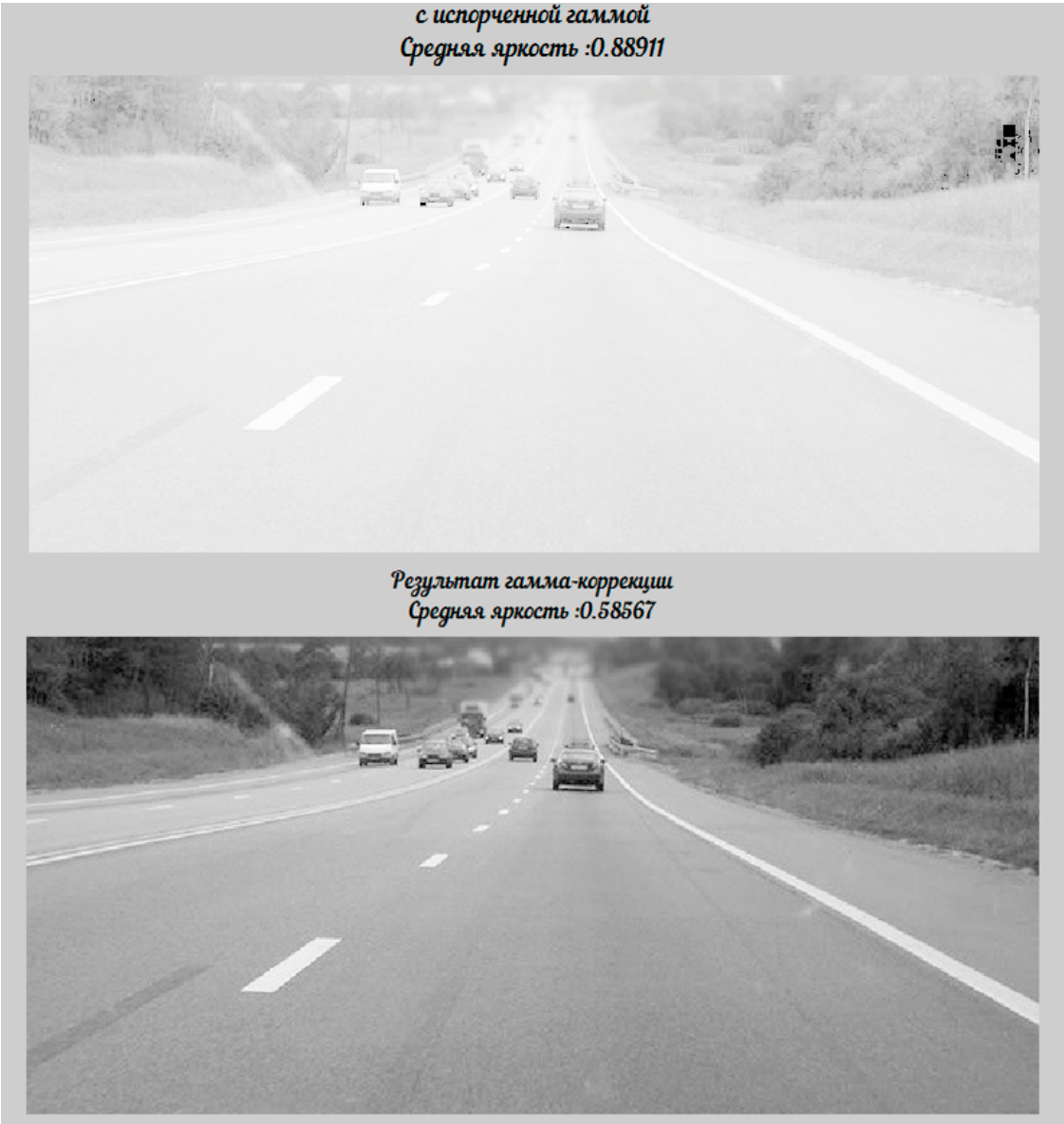


Рис. 8. Результат применения гамма-коррекции

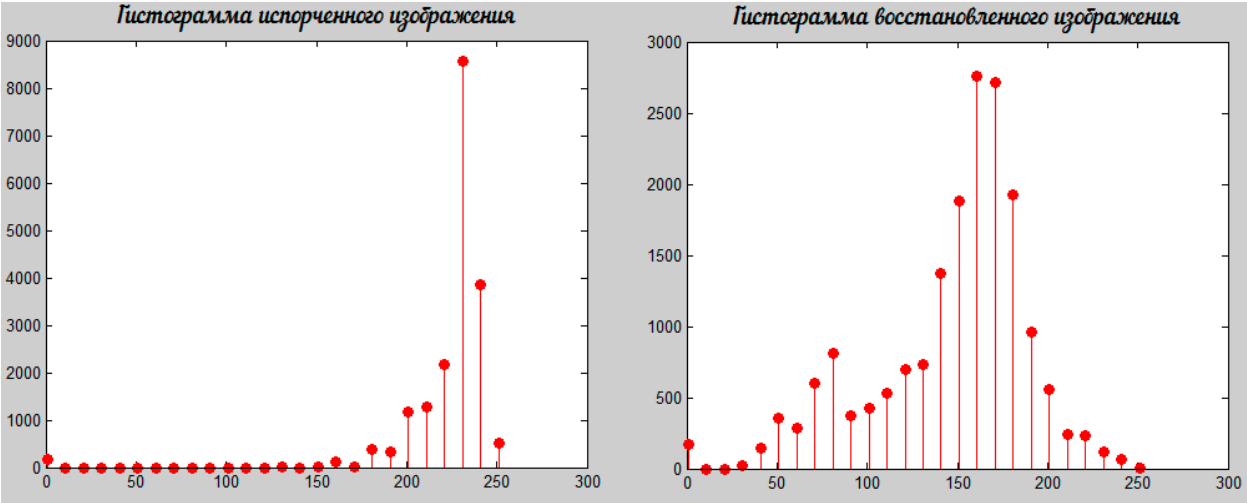


Рис. 9. Соответствующие гистограммы изображений

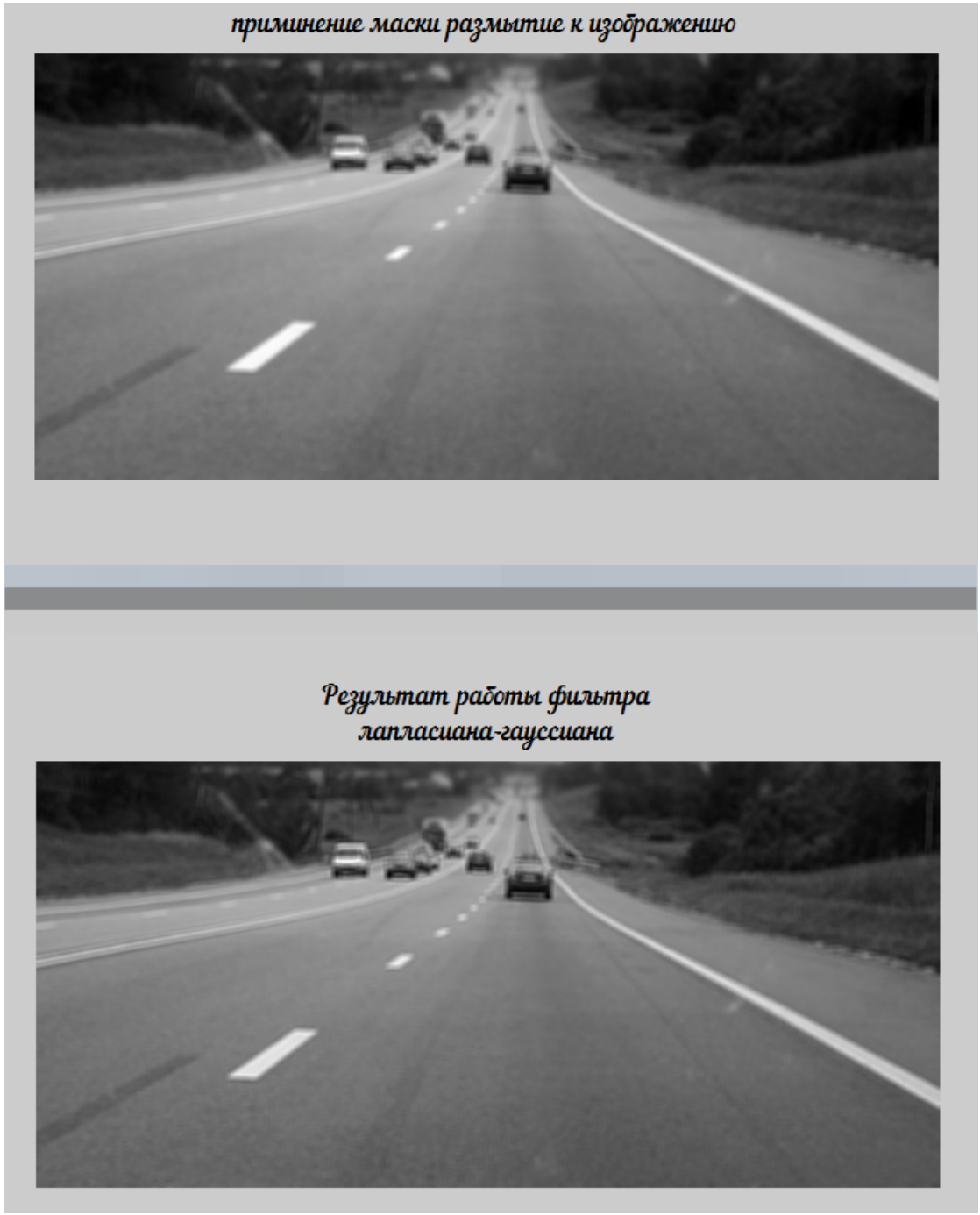


Рис. 10. Результат работы фильтра лапласиана-гауссиана

Рис. 11. Результат повышения контрастности

Рис. 12. Результат применения бинаризации

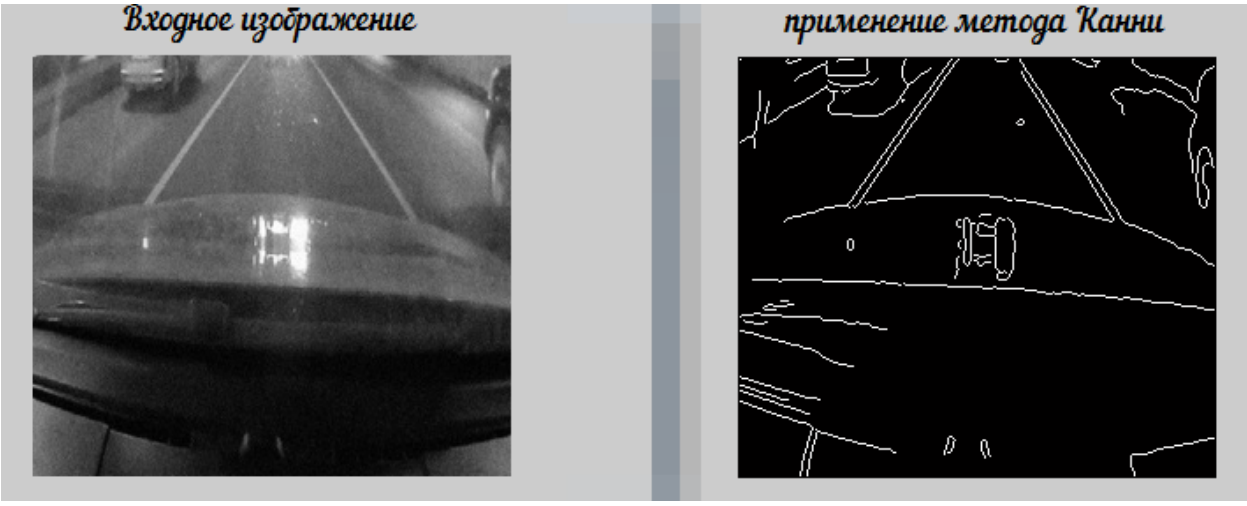


Рис. 13. Применение метода Канни к изображению

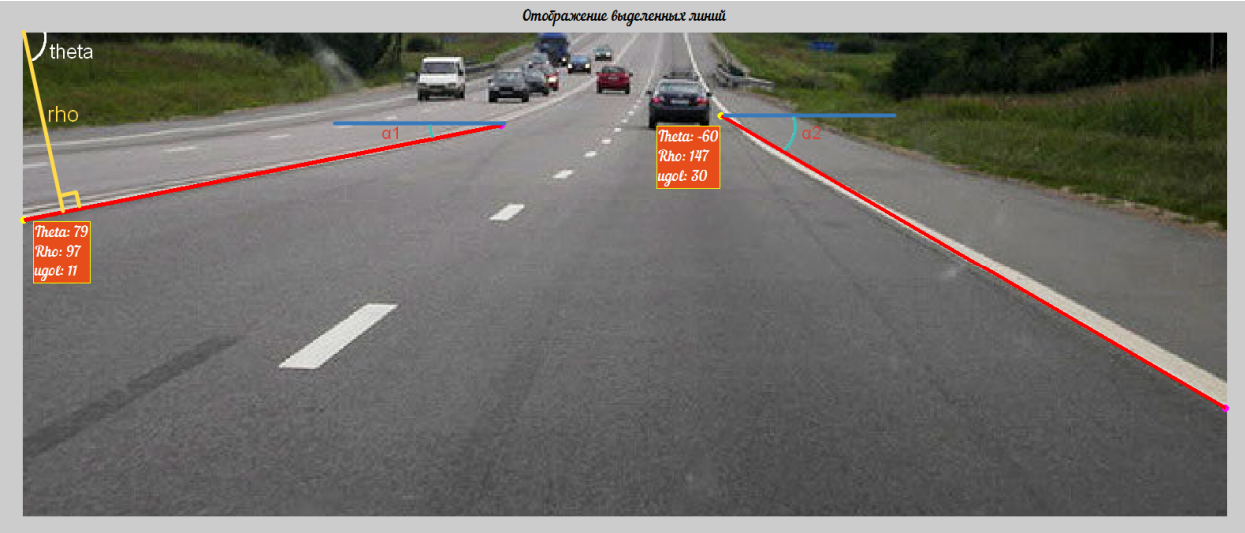


Рис. 14. Результат выделения линий дорожной разметки

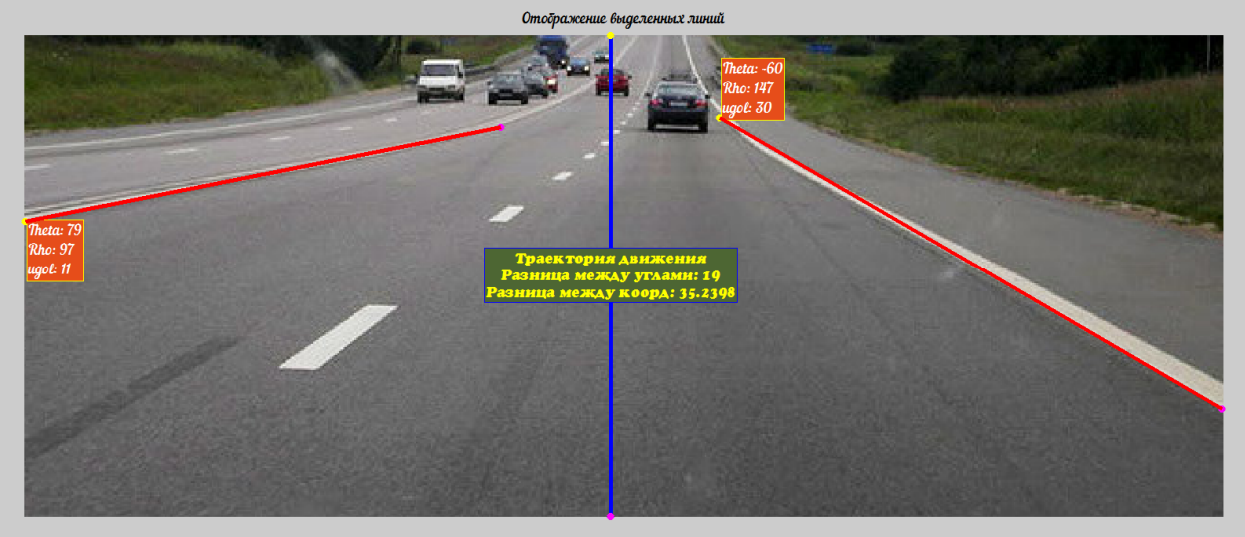


Рис. 15. Пример верного определения траектории движения транспортного средства без пересечения дороги

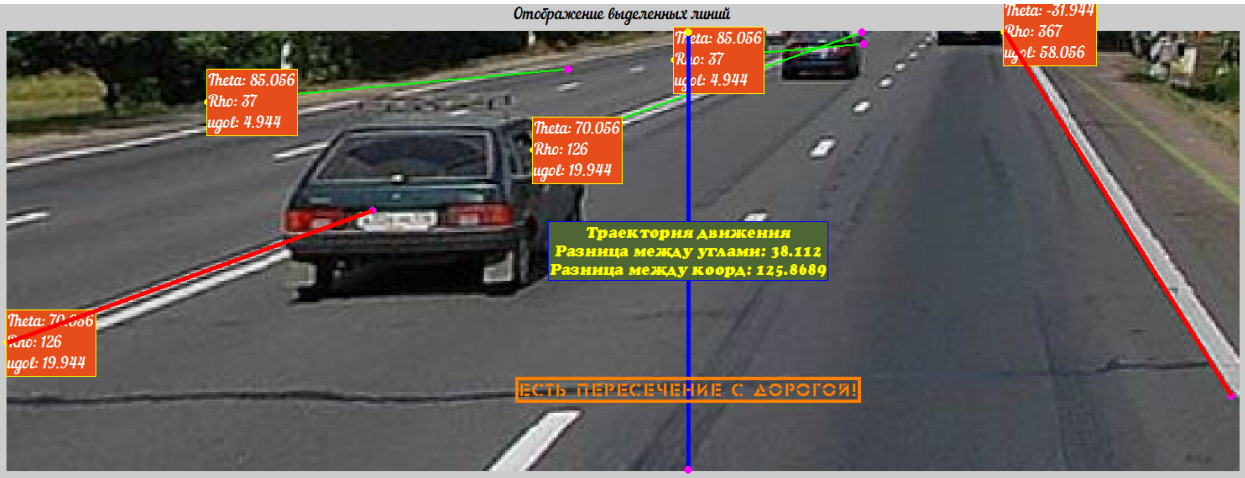


Рис. 19. Пример верного определения траектории движения транспортного средства с пересечением дороги

Список литературы

1. MathWorksfspecial. Available at: http://www.mathworks.com/help/images/ref/fspecial.html(датаобращения 05.12.2014)
2. MathWorks wiener2. Available at: http://www.mathworks.com/help/images/ref/wiener2.html(датаобращения 05.12.2014)
3. Контрастирование с гамма-коррекцией. Режим доступа: http://matlab.exponenta.ru/imageprocess/book3/10/imadjust.php(дата обращения 05.12.2014)
4. Википедия. Ошибки первого и второго рода. Режим доступа: https://ru.wikipedia.org/wiki/%CE%F8%E8%E1%EA%E8_%EF%E5%F0%E2%EE%E3%EE_%E8_%E2%F2%EE%F0%EE%E3%EE_%F0%EE%E4%E0(дата обращения 05.12.2014)
5. Википедия. Преобразование Хафа. Режим доступа: https://ru.wikipedia.org/wiki/Преобразование_Хафа(дата обращения 05.12.2014)
6. Википедия. Прямая. Режим доступа: https://ru.wikipedia.org/wiki/%CF%F0%FF%EC%E0%FF(дата обращения 05.12.2014)
7. Гонсалес Р., Вудс Р. Цифровая обработка изображений. М.: Техносфера, 2010. 615 с.
8. Машков К.Ю., Рубцов В.И., Рубцов И.В. Состав и характеристики мобильных роботов. М.: МГТУ им. Н.Э. Баумана, 2014. 73 с.