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

 # 07, июль 2017

УДК 004.93

Обнаружение движущихся объектов методом вычитания фона

***Походня И. С.**, студент*
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Системы обработки информации и управления»

Научный руководитель: Терехов В.И., к.т.н, доцент
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана
кафедра «Системы обработки информации и управления»
terekchow@bmstu.ru

Введение

В нашей жизни все большую часть занимают информационные технологии, которые повсеместно проникают в нашу жизнь. В результате стремительного развития технологий появилась техническая возможность реализации такого актуального направления как анализ видеопотока с целью выявления необходимой информации в реальном режиме времени.

Целью статьи является решение задачи обнаружения в видеопотоке движущихся объектов. Анализ таких объектов широко применяется в охранных системах для обнаружения объектов на охраняемой территории, в анализе дорожного траффика с целью обнаружения нарушителей скоростного режима, правил дорожного движения и т.д.

Существует несколько методов обнаружения движущихся объектов:

1. Метод оптического потока (англ. *optical flow*).
2. Метод вычитания фона (англ. *background subtraction*).
3. Метод временной разницы (англ. *temporal differencing*).

Метод оптического потока [1] определяет движение пикселей между двумя кадрами, для чего выполняет большое количество сложных вычислений. Метод требует для своей реализации большие вычислительные мощности, что является минусом этого метода.

Суть метода вычитания фона [1] состоит в анализе кадров из видеопотока и определения фона и переднего плана, которым являются движущиеся объекты. Метод требует небольших вычислительных мощностей, устойчив к изменениям в окружающей

среде и точно определяет движущиеся объекты независимо от количества кадров в секунду.

Метод временной разницы [1] является простым, быстрым и хорошо адаптируемым к изменениям в окружающей среде. Метод использует вычитания текущего кадра из предыдущего, однако ошибочно определяет движущиеся объекты при большом интервале времени между кадрами.

Постановка задачи

В статье рассматривается возможность обнаружения машин на дороге методом вычитания фона. В качестве входных данных используется видеопоток со статической камеры, установленной над дорогой в солнечный день в летнее время года, передающей данные на сервер со следующими характеристиками:

- не менее 1 Гб ОЗУ;
- процессор Intel core i3 и выше;
- Java SE 6 и выше;
- OpenCV версии 2.3 и выше.

Решение

Так как окружающая среда постоянно меняется (день сменяется ночью, меняется погода и т.п.), то для обнаружения движущихся объектов необходим такой алгоритм, который мог бы учитывать все эти изменения автоматически.

Данный алгоритм подходит для решения поставленной задачи, т.к. предназначен для выделения объектов на фоне со статического источника. Алгоритм существует в различных вариациях, в статье рассмотрен алгоритм вычитания фона с помощью смеси гауссиан.

Суть алгоритма в том, что у нас имеется *N* гауссовых нормальных распределений:

$$\boldsymbol{p}(\vec{\boldsymbol{x}}|\boldsymbol{X}, \boldsymbol{BG} + \boldsymbol{FG}) = \sum_{\boldsymbol{m}=\boldsymbol{1}}^{\boldsymbol{N}} \widehat{\boldsymbol{\alpha}}_{\boldsymbol{m}}\, \boldsymbol{\eta}\left(\vec{\boldsymbol{x}}; \widehat{\vec{\boldsymbol{\mu}}}_{\boldsymbol{m}}; \widehat{\boldsymbol{\sigma}}_{\boldsymbol{m}}^{\boldsymbol{2}}\boldsymbol{I}\right),$$

где $\hat{\vec{\mu}}_1, \dots, \hat{\vec{\mu}}_N$ – математическое ожидание, $\hat{\sigma}_1^2, \dots, \hat{\sigma}_N^2$ – дисперсия, I – ковариационная матрица, $\hat{\alpha}_1, \dots, \hat{\alpha}_N$ – веса соответствующих распределений [2].

После каждого нового изображения веса распределений обновляются по формулам:

$$\hat{\alpha}_m \leftarrow \hat{\alpha}_m + \beta(o_m - \hat{\alpha}_m),$$

$$\hat{\vec{\mu}}_m \leftarrow \hat{\vec{\mu}}_m + o_m\,(\beta/\hat{\alpha}_m)\vec{\delta}_m,$$

$$\hat{\sigma}_m^2 \leftarrow \hat{\sigma}_m^2 + o_m\,(\beta/\hat{\alpha}_m)(\vec{\delta}_m^T\vec{\delta}_m - \hat{\sigma}_m^2),$$
$$\vec{\delta}_m = \vec{x} - \hat{\vec{\mu}}_m,$$

где β задает степень влияния на параметры распределений, коэффициент o_m равен 1 для ближайших гауссовых распределений. Гауссово распределение считается ближайшим, если расстояние Махаланобиса [3] меньше некоторой задаваемой в алгоритме постоянной. Соответственно если ближайших распределений нет, то добавляется новое распределение со следующими параметрами:

$$\hat{\alpha}_m = \beta,\ \hat{\vec{\mu}}_{N+1} = \vec{x},\ \hat{\sigma}_{N+1} = \hat{\sigma}_0,$$

где $\hat{\sigma}_0$ – некоторое начально значение.

Для объектов, которые находятся на изображении давно, весовые коэффициенты больше. Таким образом, для определения того, является ли пиксель фоном, достаточно взять *B* распределений с наибольшими весовыми коэффициентами:

$$\boldsymbol{p(\vec{x}|X, BG) \sim \sum_{m=1}^{B} \alpha_m\,\eta(\vec{x}; \hat{\vec{\mu}}_m; \hat{\sigma}_m^2 I)}.$$

В результате классификации пикселей выделяется передний план - участки изображения, которые появились на нем недавно. Если на изображении появится новый объект, то для него будет создан новый кластер. Вес такого кластера будет расти со временем, вследствие чего он попадет в *B* распределений, классифицирующих пиксель как фон.

Так как со временем все веса распределений обновляются, появляется устойчивость алгоритма к изменениям в окружающей среде источника, при том, что обновление фона происходит постепенно.

Результат выделения фона на дороге для монохромного изображения представлен на рис. 1.

Рис. 1. Выделение фона среди потока машин

Реализация алгоритма обнаружения движущихся объектов

Для обработки изображений использовалась библиотека *OpenCV* [4], которая имеет открытый код и распространяется под лицензией *BSD*, что допускает ее использование в работе коммерческих продуктов и для учебных целей. Сама библиотека содержит большое количество алгоритмов на широкую тематику, включая обработку изображений. Все использованные в работе скриншоты являются результатом моделирования в *3ds Max* [5].

Библиотека *OpenCV* существует для различных языков: *Java*, *Python*, *C++* и др., однако в работе использовался язык *Java* и соответствующая ему реализация библиотеки.

Для работы с видеопотоком используется класс *VideoCapture*, который позволяет получить поток как с веб-камер, так и с файлов, хранящихся в памяти. *VideoCapture* считывает изображения, используя один метод *read(Mat destination*), загружающий изображение из потока в объект типа *Mat*, и представляющий изображение в виде матрицы [6].

Для решения поставленной задачи необходимы следующие модули библиотеки *OpenCV*:

– *core* – модуль с базовой функциональностью для работы с матрицами (математические функции, основные типы объектов, генераторы случайных чисел) [7].

– *imgproc* – модуль с функционалом для обработки изображений (операции сужения, расширения, размытия, изменения размера) [7].

– *videoio* – модуль с базовым функционалом для работы с видео (чтение потока, запись потока, флаги для настройки видеопотока) [7].

Рассмотрим обработку изображения подробнее. Изначально имеется считанное из видеопотока изображение, показанное на рис. 2.

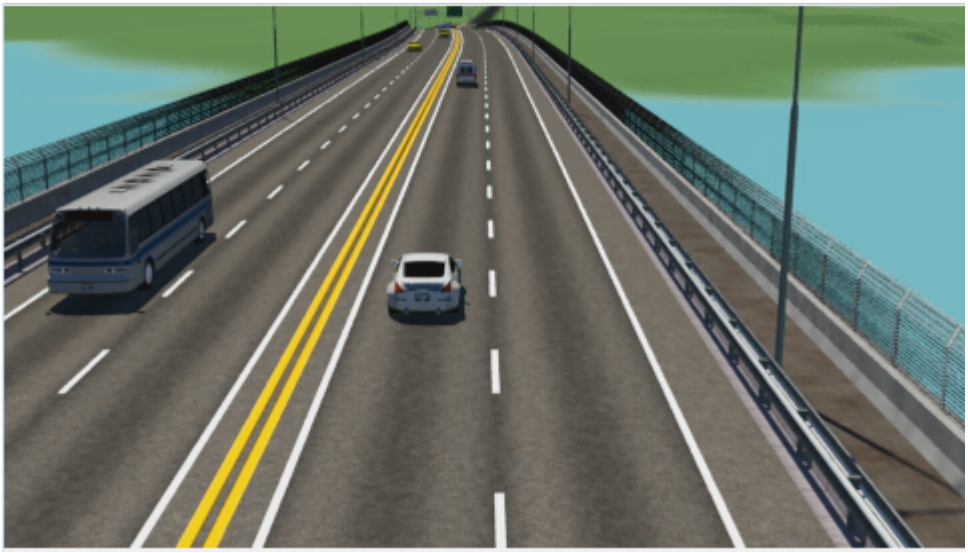

Рис. 2. Считанное из видеопотока изображение

Ниже приведен алгоритм обнаружения движущихся объектов на изображении из видеопотока, который состоит из следующих пунктов:

1. Преобразование цветного изображения в монохромное с помощью функции *Imgproc.cvtColor(…)*, где параметр, задающий преобразование изображения, равен *Imgproc.COLOR_RGB2GRAY*.

2. Применение размытия по Гауссу для изображения с целью уменьшения влияния таких незначительных деталей, как движение деревьев. Выполнение размытия по Гауссу выполняется функцией *Imgproc.GaussianBlur(…)*.

3. Применение алгоритма вычитания фона. В *OpenCV* алгоритм вычитания фона представлен в виде объекта *BackgroundSubtractorMOG2*, чтобы применить алгоритм, достаточно выполнить функцию *apply* на текущем изображении из видеопотока.

4. Применение порогового преобразования для выделения наиболее ярких участков на изображении: задается значение порога, если значение пикселей больше него, то они остаются на изображении и принимают некоторое заданное в методе значение, иначе пиксели становятся черными. В результате изображение содержит всего два цвета. Обычно в качестве значения пикселей, прошедших порог, берется значения белого цвета. Для вызова порогового преобразования используется функция *Imgproc.threshold(…)*.

5. Применение расширения изображения для укрупнения оставшихся участков, что позволяет в будущем точнее выделить движущиеся объекты. Функция проходит по

изображению маской, представляющей собой круг или квадрат с ведущей позицией в центре. В маске ищется локальный максимум, который применяется к остальным пикселям, тем самым делая изображение светлее. Для расширения используется функция *Imgproc.dilate(…).* [8] Результат предыдущих шагов представлен на рис. 4.

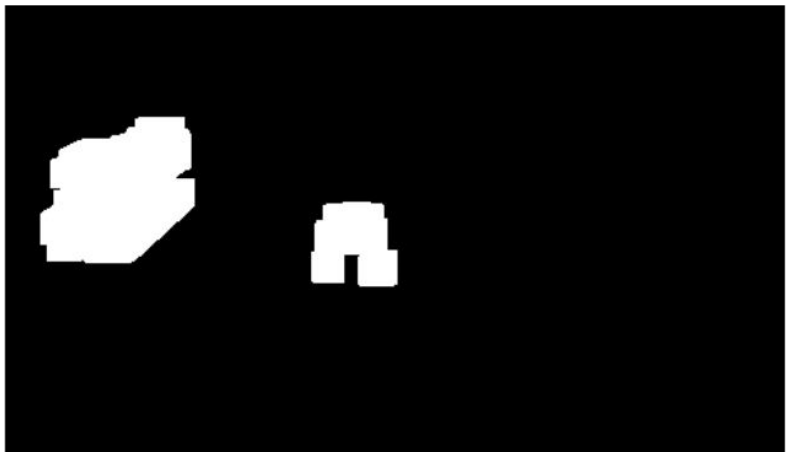

Рис. 4. Обработанное изображение после укрупнения значимых участков

6. Выполнение поиска контуров среди укрупненных участков изображения. Поиск контуров осуществляется функцией *Imgproc.findContours(…).* Стоит отметить, что поиск контуров возможен только на бинарном изображении. В рассматриваемом случае изображение становится бинарным при выполнении 4 пункта алгоритма.

7. Выделение контуров с помощью прямоугольников и отображение этих прямоугольников на изображении. Отображение прямоугольников на исходном изображении осуществляется с помощью функции *Imgproc.rectangle(…).*

Конечный результат алгоритма после отображения движущихся объектов представлен на рис. 5, а блок-схема алгоритма, на рис. 6.

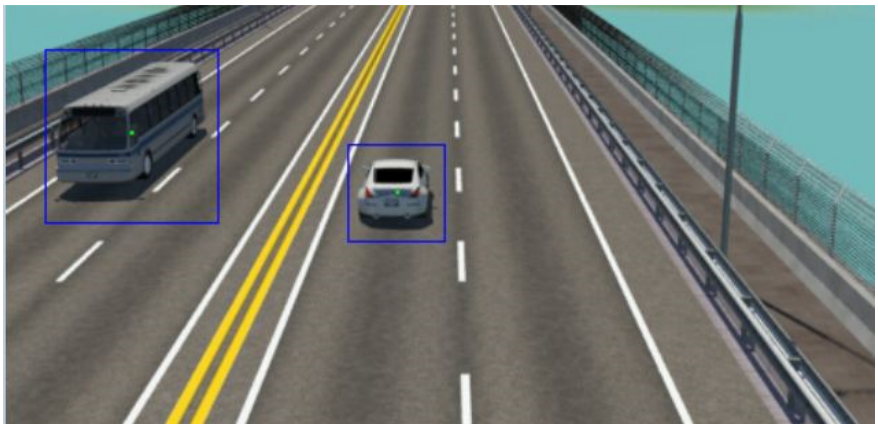

Рис. 5. Результат выполнения алгоритма поиска движущихся объектов

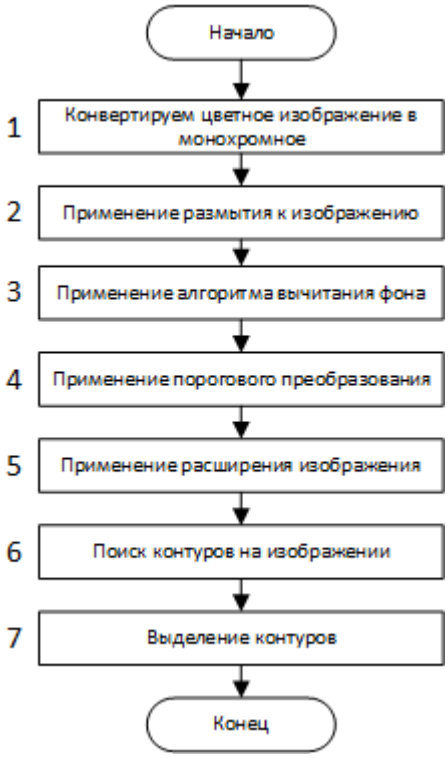

Рис. 6. Алгоритм поиска движущихся объектов

Направление дальнейших исследований

Направлением дальнейших исследований может служить решение задач обнаружения остановки машины, определения ее номера, марки, типа или скорости движения, используя метод обнаружения движущихся объектов как подготовительный этап решения перечисленных задач.

Заключение

В статье был рассмотрен способ обнаружения движущихся объектов в видеопотоке на основе библиотеки OpenCV с помощью метода вычитания фона. В результате проведенных экспериментов была показана простота работы алгоритма, его скорость и возможность приспосабливаться к изменяющимся условиям в окружающей среде источника при решении задачи обнаружения машин на дороге. В качестве недостатка выбранного алгоритма отмечается возможность его работы только с информацией из неподвижного источника данных.

Список литературы

[1]. Shaikh Soharab Hossain, Saeed Khalid, Chaki Nabendu. Moving Object Detection Approaches, Challenges and Object Tracking // Moving Object Detection Using Background Subtraction, Springer International Publishing. 2014. Ch. 2. P. 5-14.

[2]. Zoran Zivkovic and Ferdinand van der Heijden. Efficient adaptive density estimation per image pixel for the task of background subtraction // Pattern recognition letters. 2006. № 27(7). P. 773–780.

[3]. Расстояние Махаланобиса. Режим доступа: http://statistica.ru/theory/rasstoyanie-makhalonobisa/ (дата обращения 03.05.2017).

[4]. Официальный сайт OpenCV. Режим доступа: http://opencv.org/ (дата обращения 01.05.2017).

[5]. Официальный сайт Autodesk 3ds Max. Режим доступа: https://www.autodesk.ru/products/3ds-max/overview (дата обращения 13.05.2017).

[6]. Работа с OpenCV. Часть 1. Установка и Hello World. Режим доступа: https://habrahabr.ru/post/204638/ (дата обращения 11.05.2017).

[7]. Документация OpenCV. Режим доступа: http://docs.opencv.org/ (дата обращения 13.05.2017).

[8]. OpenCV шаг за шагом. Обработка изображения - морфологические преобразования. Режим доступа: http://robocraft.ru/blog/computervision/319.html (дата обращения 10.05.2017).