УДК 004.4

Е.Е. Федоров, д-р техн. наук,
Д.А. Соляр, студент
Донецкий национальный технический университет, г. Красноармейск
fedorovee75@mail.com, ggrimber@gmail.com

Разработка подсистемы автоматизации контроля трафика автостоянки

В данной работе решается проблема создания эффективных алгоритмов и программных средств распознавания графических образов в реальном времени. Разработана подсистема для автоматизации контроля трафика автостоянки, которая позволяет повысить эффективность обработки изображений. Для предложенной подсистемы создана математическая модель.

Ключевые слова: распознавание, видеопоток, камера наблюдения, фильтрация, сегментация.

Введение

В современной жизни информация играет очень важную роль. Научно-технический прогресс в областях, связанных с передачей и обработкой информации, открывает всё новые и новые горизонты. Вслед за этим быстро развивается и информатика. С ростом производительности персональных компьютеров, емкости носителей важным направлением развития информационных технологий стали средства мультимедиа. Сначала они служили в основном для украшения компьютерных игр, но со временем нашли много других применений своей мощи и растущим возможностям. Помимо игр появились сначала короткие видеоролики, аудиотреки, затем полнометражные фильмы в цифровых форматах.

Сегодня мультимедийные потоки применяются достаточно широко в различных отраслях производства и сферах жизни людей. Они играют роль очень удобного носителя постоянно поступающей информации, и поэтому их популярность стремительно растёт. Можно выделить две области, где они себя особо зарекомендовали: это видеофильмы и телеконференции. Причины этому – удобство и простота использования. Сравнительно новым способом применения потоковых мультимедийных данных стала автоматизация контроля за перемещением автомобильного транспорта. Здесь необходимо применение дополнительных алгоритмов, позволяющих не просто передавать и отображать информацию, поступающую на вход, а ещё и отслеживать изменения относительно предшествующих данных и информировать об их наличии пользователя либо вести запись истории таких изменений.

Существует две больших группы детекторов: детекторы активности и детекторы движения. Детекторы активности основаны на обнаружении активности в потоке данных. Подобная активность может быть вызвана каким-либо движущимся объектом, однако не исключены и ложные срабатывания на изменения условий освещённости или шум при регистрации и передаче данных. Детекторы движения работают по иному принципу. Их задача – реагировать именно на движение в кадре. Источником движения может служить человек, животное, либо механическое устройство, изменяющее своё положение относительно устройства регистрации данных в течение нескольких кадров.

Решение проблемы автоматизации контроля трафика автостоянки является актуальной задачей, связанной с широким классом практических приложений. Создание автоматической подсистемы, регистрирующей движение автомобилей, позволит автоматизировать контроль въезда и перемещения транспортных средств на объектах с ограниченным доступом и закрытых территориях, отслеживать въезд и выезд на автостоянках, осуществлять автоматический подсчет стоимости предоставленных услуг, контролировать свободное место, автоматизировать контроль выезда оплаченных или неоплаченных транспортных средств на станциях технического обслуживания и автокомбинатах, контролировать загрузку зоны обслуживания.

Существующие системы, такие, как, например, VideoIQ компании General Electric или Видео-Инспектор компании ISS, – это дорогостоящие продукты, поэтому их использование связано с большими затратами. Попытка реализовать аналог одной из существующих систем по литературным данным с самостоятельной доработкой наталкивается на трудности, связанные с ограниченным объемом

информации. Кроме того, известные разработки также несвободны от недостатков и не всегда могут использоваться напрямую для решения данной задачи.

Объект, цель и задачи исследования

Объектом исследования является процесс распознавания движения автомобиля на изображении с камеры видеонаблюдения, который выполняется с помощью обработки кадров изображения определенными заданными математическими алгоритмами.

Входными данными является изображение отдельных кадров с камеры видеонаблюдения, которая установлена неподвижно и направлена сверху или почти сверху на участок за въездом на автомобильную стоянку. Дополнительных ограничений на размещение не накладывается. Данные с камеры видеонаблюдения (вероятно, что это будет web - камера) поступают на обработку в систему через компьютер, который обрабатывает информацию. Входные данные с камеры подвергаются обработке с помощью алгоритмов обработки изображения, чтобы потом определить, есть ли на изображениях движущийся автомобиль и в каком направлении он движется. Система должна отслеживать движение автомобиля, направление его движения, вести подсчёт въехавших и выехавших машин, и сохранять кадры с въезжающими/выезжающими машинами в файл.

Целью данной работы является разработка подсистемы для автоматизации контроля трафика автостоянки, путём исследования и разработки методов и алгоритмов детектирования движения объектов на изображении и определения направления их движения. Что, в свою очередь, обеспечит автоматическую обработку видеоинформации с камеры наблюдения.

Для достижения поставленной цели необходимо решить следующие задачи:

- провести анализ методов детектирования движущихся объектов на изображении и определения направления их движения;
- разработать и модифицировать алгоритм детектирования движения на изображениях с камеры видеонаблюдения;
- разработать алгоритм определения направления движения объекта на изображениях с камеры видеонаблюдения;

Общие сведения

Разрабатываемая подсистема должна принимать на вход кадры видеопотока в виде растровых изображений в формате RGB, полученные с цветной или черно-белой видеокамеры либо загружаемые из видеофайла, обрабатывать эти данные, обнаруживая на них движущиеся объекты (автомобили), определять направление их движения на видеокадрах и сохранять кадры в файл.

Предварительная обработка изображения

На этапе предобработки необходимо подготовить видеоизображения к обнаружению на них движущихся объектов. Можно считать, что предобработка проведена качественно, если любые два видеокадра, полученные с небольшим интервалом от одной и той же камеры в условиях отсутствия движения в зоне видимости камеры являются максимально похожими друг на друга. Похожесть кадров затем можно определить, вычислив значение функции их взаимной корреляции или построив маску движения после вычисления межкадровой разности.

На первом этапе предобработки с целью подавления шумов выполняется сглаживание изображения с помощью фильтра Гаусса в области 3x3 вокруг каждого пикселя изображения. Матрица свертки в данном случае вычисляется по закону распределения Гаусса:

$$G(x,y)=\frac{1}{(2\pi\sigma^2)}e^{\frac{(x^2+y^2)}{2\sigma^2}},$$

где: σ — стандартное отклонение распред (1
Гаусса.

На втором этапе выполняется преобразование цветного изображения в оттенки серого, по известной зависимости, отражающей свойства человеческого глаза:

Y=0.3*R+0.59*G+0.11*B, (2)

где R,G,B – значения яркости соответственно красного, зелёного и синего цвета (рис. 1).

Рисунок 1 – Исходное изображение и изображение после предобработки

Определение движения объекта на изображении

После завершения этапа предобработки кадров с целью подготовки их к обнаружению движения, можно приступить к выполнению последовательности действий, включающей вычисление межкадровой разности, формирование бинарной маски движения, обнаружение движения, выделение и трассировку объектов. Алгоритм вычисления межкадровой разности двух кадров для случая обработки кадров видео в формате RGB выглядит следующим образом:

1) На вход алгоритма поступают два видео кадра, представляющие собой две последовательности байт в формате RGB.

2) Производится вычисление попиксельных межкадровых разностей по следующей схеме:

$$R_{res}^i = |R_1^i - R_2^i|$$
$$G_{res}^i = |G_1^i - G_2^i| \quad (3)$$
$$B_{res}^i = |B_1^i - B_2^i|$$

где R_{res}^i, G_{res}^i, B_{res}^i - значения красной, зелёной и синей компонент цвета *i*-го пикселя результирующего растра, $R_1^i, R_2^i, G_1^i, G_2^i, B_1^i, B_2^i$ - значения красной, зелёной и синей компонент цвета i-го пикселя на первом и втором кадре.

3) Для каждого пикселя вычисляется среднее значение между значениями трёх компонент цвета:

$$p^i = \frac{R_{res}^i + G_{res}^i + B_{res}^i}{3} \quad (4)$$

4) Среднее значение сравнивается с заданным порогом. В результате сравнения формируется двоичная маска:

$$m_i = \begin{cases} 0, p^i < T \\ 1, p^i \geq T \end{cases} \quad (5)$$

где m_i – значение *i*-го элемента маски, T – порог сравнения, иногда называемый также порогом или уровнем чувствительности (рис. 2).

Таким образом, на выходе алгоритма формируется двоичная маска, одному элементу которой соответствуют три компоненты цвета соответствующего пикселя исходных двух кадров. Единицы в маске располагаются в областях, где, возможно, присутствует движение, однако на данном этапе могут быть и ложные срабатывания отдельных элементов маски, ошибочно установленных в 1.

При вычислении межкадровой разности в качестве одного из кадров берётся базовый кадр. Преимущества такого подхода подробно описаны в первой части. Среди них: точность локализации объекта на текущем кадре, обнаружение малоразмерных объектов, наличие ненулевых элементов в маске движения не по контуру, а по всей площади объекта.

По значениям межкадровой разности строится маска движения с использованием задаваемого пользователем порога. Элементы со значениями, меньшими порога, обнуляются, остальные становятся единицами в маске движения. В построенной таким образом маске движущиеся объекты оставляют чёткие следы, которые по форме достаточно сильно напоминают силуэты исходных объектов (рис 2).

Рисунок 2 - Исходное изображение и результат межкадровой разности

Работа по выделению объектов детектором движения, основанным на алгоритме вычисления попиксельных разностей между кадрами, начинается с анализа маски движения.

Реальным объектам в ней обычно соответствуют пиксели, которые образуют связные группы, поэтому логично определить объект с точки зрения детектора движения как группу связных пикселей в маске движения. Такому объекту соответствуют несколько параметров:

1) Линейные размеры минимального прямоугольника (в дальнейшем просто прямоугольник), который можно описать около пикселей группы.

2) Координаты (x, y) центральной точки прямоугольника. Будем считать данную точку центральной точкой объекта.

3) Количество пикселей, входящих в группу.

4) Область маски движения, лежащая внутри прямоугольника.

5) Область текущего кадра, лежащая внутри прямоугольника.

Поскольку детектор обнаруживает только движущиеся объекты, эта группа пикселей и соответствующий ей объект с поступлением на вход детектора новых кадров будут испытывать смещение, поэтому кроме параметров, перечисленных выше, можно ввести ещё несколько:

6) Вектор, описывающий направление и скорость движения объекта.

7) Массив, содержащий координаты центра объекта на предыдущих кадрах.

8) Время жизни объекта, измеряемое в количестве кадров.

Чтобы выделить движущиеся объекты в маске движения, осуществляется последовательный обход её пикселей с пометкой найденных ненулевых элементов. При нахождении ненулевого непомеченного элемента в маске, запускается процедура поиска остальных 8-связных с ним ненулевых элементов с помощью волнового алгоритма.

Из точки, где расположен текущий пиксель (первый найденный пиксель, которому соответствует единица в маске движения), запускается волна во всех направлениях, число которых зависит от выбранного типа связности (4-связный, 8-связный). Для всех пикселей, которые достигаются волной в текущей итерации, производится проверка условий наличия единицы в маске движения и отсутствия пометки о вхождении в другой объект. Все пиксели, успешно прошедшие проверку, заносятся в массив. На следующей итерации волна запускается из всех пикселей, содержащихся в массиве. Массив очищается, и в него заносятся пиксели, вновь достигнутые волной. Все обработанные пиксели помечаются как вошедшие в текущую группу. При реализации данного алгоритма удобно иметь массив переменной длины для помещения в него пикселей волны на текущей итерации, либо заранее вычислять максимально допустимый размер этого массива.

Обработка маски, алгоритмом волнового поиска проиллюстрирована на рис. 3. Как видно, поиск идёт одновременно во всех возможных направлениях, определяемых выбранным типом связности (здесь 8-связное соседство), что существенно сокращает количество итераций поиска и затраты памяти на хранение промежуточных результатов между итерациями. Алгоритм выполняется до тех пор, пока на очередном шаге массив содержит хотя бы один пиксель, из которого запускается новая волна.

При этом корректируются размеры обрамляющего группу прямоугольника, который изначально имеет нулевую площадь, располагаясь в точке, в которой находится исходный пиксель, а затем увеличивается в размерах вслед за включением новых пикселей в группу. С точки зрения математики прямоугольник не может иметь нулевой площади, однако нам такой подход удобен при реализации алгоритма, потому что прямоугольник в данном случае всегда будет ограничивать группу пикселей, не выходя за её пределы, что позволит избежать ошибок выхода за границу массива при работе с объектами, находящимися у края изображения.

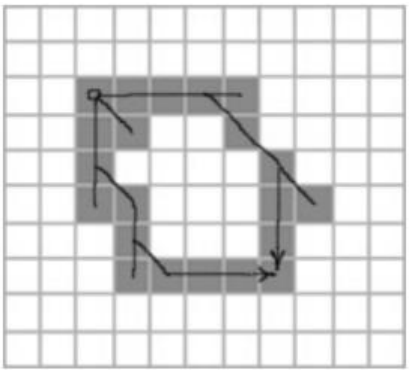

Рисунок 3 - Пример работы волнового алгоритма поиска связных групп пикселей

Пример обрамляющего прямоугольника представлен на рис. 4. Видно, что такой обрамляющий прямоугольник никогда не будет выступать за пределы растра, поскольку его границы лежат поверх крайних пикселей области, которую они окаймляют.

Рисунок 4 - Группа связных ненулевых элементов в маске движения и обрамляющий её прямоугольник

После того, как все группы связных ненулевых элементов маски обнаружены, информация о них заносится в массив объектов.

Если на входе алгоритма не первый кадр потока, то, вероятно, массив объектов уже не пуст, и необходимо найти соответствие между уже обнаруженными объектами с предыдущих кадров и объектами, найденными на текущем кадре. Такое соответствие может быть найдено путём наложения объектов из массива на текущий кадр и поиска пересечений. Наложение

проводится с учётом скорости и направления движения объекта. Объект переносится в новое положение посредством сложения его текущих координат и координат вектора скорости его движения.

$$Obj.x = Obj.x + Obj.xVect \quad (6)$$
$$Obj.y = Obj.y + Obj.yVect$$

Если соответствие найдено, значит объект с предыдущего кадра был обнаружен на текущем, и новый объект в массив не заносится. Вместо этого происходит лишь обновление параметров уже имеющегося объекта. Корректируется его текущее положение, размер, скорость и направление движения. Если соответствие не было найдено, значит, объект, вероятно, исчез за каким-либо препятствием. Такой объект остаётся в массиве объектов, однако помечается как исчезнувший. Теперь для всех вновь обнаруженных объектов будет вычисляться их похожесть на этот исчезнувший объект, пока либо не будет найдено сходство, что могло бы означать, что скрывшийся объект появился вновь, либо не истечёт интервал времени, после которого скрывшийся объект считается исчезнувшим и удаляется из массива. Этот интервал был выбран равным 200 кадрам, чего вполне достаточно для непрерывного трассирования объекта, исчезнувшего даже за большим препятствием. По завершении обработки объектов текущего кадра, все изменения на текущем кадре, кроме тех, которые вызваны движущимися объектами, фиксируются в базовом кадре.

Это происходит путём попиксельного обновления областей базового кадра, в которых движущиеся объекты не были обнаружены, пикселями текущего кадра.

Здесь же необходимо сказать о простой коррекции, используемой при обновлении базового кадра и позволяющей уменьшить реакцию алгоритма на покачивание камеры и скачки яркости исходного изображения, возможные при включении/выключении искусственного освещения, при перемене погоды.

Идея заключается в том, чтобы корректировать пиксели базового кадра таким образом, чтобы эта коррекция минимизировала разницу между значениями пикселей текущего и базового кадра. Такой подход позволяет не реагировать на качание камеры и резкие скачки яркости, связанные с переменной освещения, которые редко длятся в течение большого количества кадров, и затем быстро гасятся в базовом кадре с помощью данного преобразования.

Локализация движущегося объекта с помощью признаков Хаара

После завершения этапа определения движения объекта на изображении, можно приступить к локализация движущегося объекта. Точная локализация движущегося объекта, в данном случае – автомобиля, на изображении выполняется с помощью признаков Хаара по методу Виолы-Джонса [19], который позволяет обнаруживать объекты на изображениях в реальном времени. Признаки Хаара представляют собой двоичную аппроксимацию вейвлета Хаара. Каждый признак представляет собой двоичную маску, т.е., черно-белое изображение, как можно видеть на Рис. 5

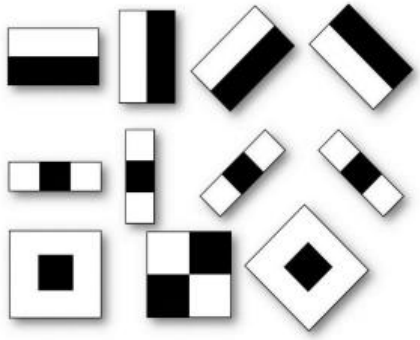

Рисунок 5 - Группа связных ненулевых элементов в маске движения и обрамляющий её прямоугольник

В данной реализации алгоритма признаки представлены набором прямоугольников. Для каждого прямоугольника задан вес, который вычисляется как сумма значений яркости пикселей, закрываемых данным прямоугольником. Такое представление обусловлено тем, что возможно быстро вычислять сумму яркостей для прямоугольных областей, но не для фигур произвольной ориентации. Признаки группируются в этапы (stages). Для каждого этапа в каскаде хранится пороговое значение (threshold). Для каждого признака хранятся два критических значения left_val и right_val.

Алгоритм использует готовую базу признаков для обнаружения объектов. Базу возможно сгенерировать из всех возможных комбинаций признаков Хаара путем отсеивания «слабых» классификаторов, которые дают ошибку второго рода, то есть ошибочно не находят объект на изображении. Один из способов получения базы с применением алгоритма AdaBoost описан в [19]. В данной работе этап генерации базы не применяется, и реализуется только этап обнаружения объекта с применением готового набора признаков и функции *cvHaarDetectObjects* библиотеки OpenCV. В качестве базы используется каскад

cars3.xml из проекта SURF-2001 Калифорнийского технологического института [20].

Каскад признаков состоит из нескольких стадий (stages). Каждая стадия включает в себя набор признаков (features). В представлении, использующемся в каскаде из проекта SURF, признаки разбиваются на одноцветные прямоугольники (rects), каждому из которых назначен положительный или отрицательный вес.

Во время выполнения алгоритма, "окно" размером Wh* Ww пикселей движется вдоль изображения по горизонтали и вертикали. Начальный размер окна выбирается равным размеру окна, записанному в каскаде-классификаторе. На каждом шаге размер окна увеличивается. При этом, размер окна на каждой итерации умножается на масштабирующий коэффициент ITER_SCALE. Окно сдвигается по горизонтали на $w_{шаг\ x}$ пикселей, и на $w_{шаг\ y}$ пикселей по вертикали.

Данные переменные вычисляются следующим образом:

$$w_{шаг\ x} = max(1, min(4, W_w/10))$$

$$w_{шаг\ y} = max(1, min(4, W_h/10))$$

Для окна высчитывается нормировочный коэффициент.

$$\mathbf{k_{норм.}} = \mathbf{1.}\frac{\mathbf{0}}{\mathbf{W_h * W_w}}$$

Внутри окна считается сумма яркостей пикселей $S_{ярк.}$ и сумма квадратов яркостей $S_{кв.}$ пикселей с использованием интегрального представления изображений. Считается математическое ожидание яркости пикселя $M_{п.}$, дисперсия $D_{п.}$ и стандартное отклонение

$$M_{п.} = S_{ярк.} * k_{норм.}$$

$$D_{п.} = (S_{кв} * k_{норм.}) - M_{п.}$$

$$\sigma = \begin{cases} \sqrt{D_{п.}}, \text{если } D_{п.} \geq \mathbf{0} \\ \mathbf{1.0\ иначе} \end{cases}$$

Для оптимизации введена пороговая константа MIN_SIZE, и если размер окна меньше этого значения, данное "окно" пропускается. Это сделано для того, чтобы исключить области, где отсутствует гарантии нахождения объекта.

Для каждой стадии из каскада считается накопленный коэффициент S_{stage}, и сравнивается с критическим значением, записанным в каскаде. В том случае, если значение меньше критического, стадия считается невыполненной, и для текущего окна обработка прерывается. В случае, если все стадии отработали корректно, считается, что в текущем окне обнаружен объект (автомобиль). При этом, введена проверка, что размер окна не больше MAX_SIZE, что позволяет регулировать порог срабатывания.

Для каждого признака из каскада считается нормированное значение S_{feat}*$k_{норм}$ и сравнивается с нормированным критическим значением, записанным в каскаде $feature \rightarrow threshold$ * σ. В случае, если значение меньше критического, к значению стадии прибавляется значение из левого поддерева признака, в противном - из правого. Значение S_{feat} считается как сумма яркостей пикселей, которые попадают в прямоугольную область внутри признака, умноженная на вес области. Координаты области умножаются на ITER_SCALE.

В данной работе используются следующие значения констант, подобранные экспериментально:

ITER_SCALE 1.1

MIN_SIZE 100x100

MAX_SIZE - по размеру изображения.

Для каждой из областей, потенциально содержащих автомобиль, которые были обнаружены с помощью данного способа, производится проверка на наличие внутри них групп движущихся пикселей. Если для какой-либо области не нашлось такой группы, то данная область удаляется из списка и считается, что объект в данной зоне не обнаружен.

Численное исследование

Тестирование разработанной подсистемы проводилось с использованием ряда видеороликов, подготовленных в ходе работы с помощью цифровой видеокамеры в реальных условиях, близких к тем, в которых работает типичная система внешнего наблюдения. При составлении тестового набора видеороликов учитывалось требования о наличии в наборе роликов:

1) Снятых в различных окружающих условиях, включая различное время суток, погодные условия.

2) С различным разрешением видеокадров.

3) Полученных с различных цветных и чёрно-белых камер.

В итоге был составлен набор из 8 видеороликов, В табл. 1 приведено короткое описание и характеристики каждого ролика из набора. Разработанный детектор показал уверенную работу на всех роликах тестового набора. Было продемонстрировано хорошее качество обнаружения объектов, независимо от условий съёмки и типа регистрирующего устройства. Не наблюдалось ложных срабатываний детектора на шум аппаратуры, лёгкое качание камеры, колебания веток деревьев, снег. Движущиеся объекты (автомобили) очерчивались прямоугольными рамками.

Таблица 1 – Ролики тестового набора

Название	Тип видео	Условия съемки	Что на видео
test1.mp4	MPEG4 Video (H264) 640x480 30 к/с, цветн.	лето, день, ясно	автомобили
test2.avi	DivX 6 640x360 25к/с, цветн.	лето, день, пасмурно	автомобили, пешеходы
test3.mp4	MPEG4 Video (H264) 640x360 25к/с, цветн.	лето, ночь, искусственное освещение	автомобили, пешеходы
taxi.mpg	MPEG1 Video 256x190 30к/с, ч/б	весна, день пасмурно	автомобили
rheinhafen.mpg	MPEG1 Video 688x564 30к/с, ч/б	весна, день, ясно	автомобили
dtneu_winter.mpg	MPEG1 Video 768x576 30к/с	зима, день, пасмурно	автомобили, пешеходы
dtneu_schnee.mpg	MPEG1 Video 768x576 30к/с	зима, день, пасмурно	автомобили
dtneu_nebel.mpg	MPEG1 Video 768x576 30к/с	осень, день, пасмурно, туман	автомобили

Тестирование проводилось на платформе с процессором Intel Core i5 3.2GHz под управлением операционной системы Microsoft Windows 7. Работа на роликах с разрешением 320x240 точек показывала 10-15% загрузку процессора при обработке видео на скорости 30 кадров в секунду. Обработка роликов с разрешением 640x480 точек загружала процессор на 20-30% при тех же 30 кадрах в секунду.

В данной работе для оценки качества детектирования движущихся транспортных средств используются два показателя. Согласно работам [21-23] видео рассматривается как единый набор изображений и применяются количественные показатели, которые вычисляются на основании информации о числе правильно и неправильно детектированных объектов:

– истинно положительный показатель (the True Positive Rate [21] или detection rate [22]) – отношение количества правильно продетектированных транспортных средств к общему числу транспортных средств;

– показатель числа ложных срабатываний (the False Detection Rate) – отношение количества ложных срабатываний к общему числу срабатываний детектора.

При этом считается, что объект детектирован правильно, если перекрытие детектированного $bbox_{detect}$ и размеченного $bbox_{marked}$ окаймляющего прямоугольников превышает некоторое пороговое значение T:

$$\frac{S(bbox_{detect} \cap bbox_{marked})}{S(bbox_{detect} \cup bbox_{marked})} = T$$

Если перекрытие составляет более 50%, то принимается, что объект продетектирован корректно.

В таблице 2 приведено сравнение результатов работы комбинированного алгоритма, используемого в данной работе, с двумя другими широко распространенными алгоритмами детектирования движущихся объектов, рассмотренными в главе 1 – методом межкадровой разности и методом оптического потока.

Анализируя полученные результаты можно заключить, что применение в данной работе комбинированного метода детектирования с использованием межкадровой разности и метода Виолы-Джонса позволило значительно снизить количество ложных срабатываний детектора, о чем говорит значение показателя FDR. При этом очевидно прослеживается увеличение средней точности детектирования TPR по сравнению с другими рассмотренными методами. Средняя точность детектирования составила 73%.

Таблица 2 – Сводная таблица результатов детектирования

Алгоритм	Название тестового видео	Значение показателя TPR	Значение показателя FDR
Межкадровая разность	test1.mp4	0.596	0.337
	test2.avi	0.522	0.605
	test3.mp4	0.597	0.658
	taxi.mpg	0.489	0.723
	rheinhafen.mpg	0.602	0.722
	dtneu_winter.mpg	0.438	0.532
	dtneu_schnee.mpg	0.589	0.604
	dtneu_nebel.mpg	0.678	0.478
Метод оптического потока	test1.mp4	0.744	0.167
	test2.avi	0.703	0.523
	test3.mp4	0.589	0.487
	taxi.mpg	0.632	0.336
	rheinhafen.mpg	0.690	0.387
	dtneu_winter.mpg	0.720	0.365
	dtneu_schnee.mpg	0.678	0.401
	dtneu_nebel.mpg	0.730	0.403
Межкадровая разность + метод Виолы-Джонса	test1.mp4	**0.898**	**0.0**
	test2.avi	**0.723**	**0.260**
	test3.mp4	**0.709**	**0.137**
	taxi.mpg	**0.717**	**0.247**
	rheinhafen.mpg	**0.657**	**0.240**
	dtneu_winter.mpg	**0.712**	**0.278**
	dtneu_schnee.mpg	**0.642**	**0.104**
	dtneu_nebel.mpg	**0.735**	**0.189**

Выводы

Целью настоящей работы была разработка подсистемы для автоматизации контроля трафика автостоянки, путём исследования и разработки методов и алгоритмов детектирования движения объектов на изображении и определения направления их движения, что, в свою очередь, обеспечивает автоматическую обработку видеоинформации с камеры наблюдения.

В рамках данной работы получены следующие научно-практические результаты:

– для детекторов движущихся объектов разработаны алгоритмы обработки видеоизображений и обнаружения движущихся объектов в потоке видеокадров;

– тестирование созданных алгоритмов проводилось на тестовом наборе из 8 видеороликов, на различных цифровых камер и в широком диапазоне внешних условий. В результате тестирования установлено, что разработанные алгоритмы позволили значительно сократить число ложных срабатываний детектора по сравнении с другими широко распространенными методами;

– на базе предложенных алгоритмов разработана компьютерная система обработки изображения с камеры видеонаблюдения автостоянки, позволяющая отслеживать движение автомобилей, направление движения, вести подсчёт въехавших и выехавших со стоянки машин.

Разработанная компьютерная система может быть модифицирована с целью расширения её функциональности и улучшения характеристик для охранной системы автостоянки. В этом случае, помимо обнаружения автомобиля, программа может определять его тип или, по крайней мере, давать ответ на вопрос, фиксировался ли данный автомобиль ранее. Важную роль для повышения эффективности компьютерной системы играет также фиксация и распознавание номерного знака въезжающего или выезжающего автомобиля.

Список использованной литературы

1. Гонзалес Р. Цифровая обработка изображений / Р. Гонзалес, Р. Вудс. – М.: Техносфера, 2006. – 1072 с.
2. Цифровая обработка изображений в информационных системах: учеб. пособие / И.С. Грузман, В.С. Киричук, В.П. Косых и др. – Новосибирск: Изд-во НГТУ, 2003. – 352 с.
3. Сато Ю. Обработка сигналов. Первое знакомство; 2-е издание / Ю. Сато. – М.: Додэка XXI, 2009. – 176 с.
4. Оппенгейм А. Цифровая обработка сигналов / А. Оппенгейм, Р. Шафер. – [2-е изд.]. – М.: Техносфера, 2007. – 856 с.
5. Лайонс Ричард. Цифровая обработка сигналов / Лайонс Ричард. – [2-е изд.]. – М.: ООО Бином-Пресс, 2006. – 656 с.
6. Сергиенко А.Б. Цифровая обработка сигналов / А.Б. Сергиенко. – СПб.: Питер, 2007. – 752 с.
7. Фисенко В.Т. Компьютерная обработка и распознавание изображений: уч. пособ. / В.Т. Фисенко, Т.Ю. Фисенко. – СПб.: СПбГУ ИТМО, 2008. – 192 с.
8. Яне Б. Цифровая обработка изображений / Б. Яне. – М.: Техносфера, 2007. – 584 с.
9. Шапиро Л. Компьютерное зрение / Л. Шапиро, Дж. Стокман. – М.: БИНОМ. Лаборатория знаний, 2006. – 752 с.
10. Tudan Li. CVOnline: Motion and time sequence Analysis [Электронный ресурс]. – 2002. URL.http://homepages.inf.ed.ac.uk/rbf/CVonline/motion.htm
11. Вежневец В. Введение в Computer Vision [Электронный ресурс] / В. Вежневец. – 2013. URL. http://cgm.computergraphics.ru/content/view/20
12. Horn B.K. Determining optical flow / B.K. Horn, B.G. Schunck // Artificial Intelligence. – 1981. – Vol. 17. – P. 185-203.
13. Barron J.L. Performance of optical flow techniques / J.L. Barron, J.D. Fleet, S.S. Beauchemin // International Journal of Computer Vision. – 1994. – Vol. 12, №1. – P. 43-77.
14. Anandan P. A Computational framework and an algorithm for the measurement of visual motion / Anandan P. // International Journal of Computer Vision. – 1989. – Vol. 2. – P. 283-310.
15. Singh A. Optic Flow Computation: A Unified Perspective / A. Singh // IEEE Computer Society Press. – 1991. – P. 168-177.
16. http://www.pronet.ua/rus/opyt/prom
17. http://vmc-id.com/print_product_info.php?products_id=15
18. http://vismart.biz/production/kontrol-avtovyezdov
19. Jones, M., Viola, P. (2001) Robust Real-Time Face Detection [Электронный ресурс] // International Journal of Computer Vision, 57(2), 137-154, 2004. URL: http://www.vision.caltech.edu/html-files/EE148-2005-Spring/pprs/viola04ijcv.pdf
20. http://lars.mec.ua.pt/public/Media/HaarFeatures_RoadFilms/HaarFeaturesTests/CarsRear/
21. Sivaraman S., Trivedi M.M. A General Active Learning Framework for On-Road Vehicle Recognition and Tracking // IEEE Transactions on intelligent transportation systems. 2010. V.11. №2. P.267–276.

Надійшла до редакції 10.01.2015

Є.Є. ФЕДОРОВ, Д.А. СОЛЯР
Донецький національний технічний університет, м. Красноармійськ

РОЗРОБКА ПІДСИСТЕМИ АВТОМАТИЗАЦІЇ КОНТРОЛЮ ТРАФІКУ АВТОСТОЯНКИ

У даній роботі вирішується проблема створення ефективних алгоритмів і програмних засобів розпізнавання графічних образів в реальному часі. Розроблена підсистема для автоматизації контролю трафіку автостоянки, яка дозволяє підвищити ефективність обробки зображень. Для запропонованої підсистеми створено математичну модель.

Ключові слова: розпізнавання, відеопотік, камера спостереження, фільтрація, сегментація.

Ye. FEDOROV, D.A. SOLIAR
Donetsk National Technical University, Krasnoarmiysk, Ukraine

DEVELOPMENT OF A SUBSYSTEM FOR AUTOMATION OF TRAFFIC CONTROL PARKING LOTS

In this paper, we solve the problem of creating efficient algorithms and software for pattern recognition in real time. Subsystem is designed to automate the control of traffic parking lots, which improves the efficiency of image processing. For the proposed subsystem we created a mathematical model.

Keywords: pattern recognition, video stream, surveillance camera, filtering, segmentation.