РАЗРАБОТКА СИСТЕМЫ АВТОМАТИЧЕСКОГО ДЕТЕКТИРОВАНИЯ ЛЮДЕЙ В ВИДЕОПОТОКЕ

Марков А. Р.

Аннотация

В докладе освещается возможность применения технологий компьютерного зрения для решения задач в сфере видеонаблюдения. Описывается процесс разработки интеллектуальной системы детектирования людей в видеопотоке**,** осуществляющей обработку видео с нескольких камер, подсчет проходящих людей, и ведущей запись получаемых видеоданных. В докладе представляются такие инструменты разработки как библиотека OpenCV, библиотека FFMPEG, каскадные классификаторы признаков.

Ключевые слова: система компьютерного зрения; OpenCV; C++; FFMPEG; детектирование движения; каскадные классификаторы признаков; локальные бинарные шаблоны; оптический поток.

Компьютерное (техническое) зрение – технология создания искусственных систем, основанных на использовании информации, получаемой из различных изображений. Сферами применения компьютерного зрения являются медицина, оборонная промышленность, производственная промышленность, транспорт. Другой важной областью применения компьютерного зрения является видеонаблюдение. Создание автоматизированных систем видеонаблюдения обосновано тем, что позволяет значительно уменьшить затраты человеческого труда на ведение наблюдения за территорией, а также повысить эффективность процесса и исключить ошибки, вызванные человеческим фактором.

В качестве основы для создания такой системы использован язык C++ и библиотека OpenCV. Данная библиотека, имеющая открытый исходный код, представляет собой набор алгоритмов компьютерного зрения и обработки изображений, разработанный на C/C++. Данная библиотека также является стандартным интерфейсом при создании приложений в области компьютерного зрения [1].

Важным элементом системы видеонаблюдения является возможность детектирования людей в кадре для дальнейшего использования этой информации, например для уведомления охранника или для подсчета прошедших людей за определенный промежуток времени. Эту информацию можно использовать для сбора статистики о посещаемости объектов.

В качестве инструмента для решения данной задачи рассмотрены системы, основанные на использовании сверточных нейронных сетей или на использовании каскадных классификаторов признаков. Важным требованием к разрабатываемой системе наблюдения является возможность её работы в режиме реального времени. Таким образом высокая скорость интеллектуальной обработки кадров выступает ключевым фактором. По этой причине принято решение использовать каскадные классификаторы признаков вместо нейронных сетей, так как первые показывают значительно более высокую скорость работы и позволяют применять их в системах реального времени.

В задачах классификации классификатором называется функция, выносящая решение, к какому из заранее известных классов относится исследуемый объект. В нашем случае есть лишь один заранее известный класс людей, и задача классификаторы принимать решение относится ли объект к этому классу или нет.

Для решения сложных задач классификации используется технология бустинга. Бустинг — комплекс методов, способствующих повышению точности аналитических моделей. Его принцип заключается в улучшении простых классификаторов для достижения необходимого уровня точности. Каскады классификаторов являются одним из примеров технологии бустинга. Ее суть заключается в использовании совокупности классификаторов, применяемых последовательно. При этом каждый следующий классификатор использует информацию, полученную предыдущими.

В качестве признаков, являющихся основой для классификатора рассмотрены примитивы Хаара и локальные бинарные шаблоны. Каскадные классификаторы, основанные на применении локальных бинарных шаблонов, превосходят в скорости работы свои аналоги, основанные на примитивах Хаара. Так как важным критерием разрабатываемой системы является быстродействие, в качестве основы для классификаторов выбраны локальные бинарные шаблоны.

Основным принципом локальных бинарных шаблонов является использование в качестве характеристики для пикселя значений окружающих его пикселей, и последующее вычисление средних значений этих характеристик в некоторой области [2]. На рисунке ниже приведены основные локальные бинарные шаблоны, используемые для поиска на изображениях (рис. 1).

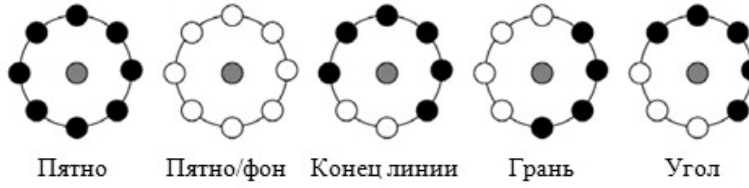

Рис. 1. Локальные бинарные шаблоны

Для создания каскадного классификатора использован функционал библиотеки OpenCV. Предварительно подготовлены обучающие наборы правильных и ошибочные примеров. После этого к правильным примерам с помощью инструмента opencv_createsamples применены автоматические изменения наклона, яркости и фона для расширения объема обучающей выборки. Следующим этапом с помощью инструмента opencv_traincascade и подготовленной обучающей выборки проводилось обучение каскадного классификатора с различными параметрами обучения (размеры изображений, объем и соотношение выборки, количество этапов обучения, требуемая точность на каждом этапе, для достижения необходимых критериев классификатора) [3]. В процессе работы над получением каскадного классификатора, обладающего необходимой точностью и скоростью детектирования объектов, обучено более пятнадцати каскадных классификаторов с различными параметрами.

Для повышения быстродействия разрабатываемой системы, в качестве инструмента для получения и записи видео использована библиотека FFMPEG. Данная библиотека, являющаяся проектом с открытым исходным кодом, предоставляет разработчику обширный функционал для работы с видео, позволяющий оптимизировать алгоритм работы программы, путем детальной настройки процесса взаимодействия с видео [4].

Учитывая необходимость максимального повышения производительности разрабатываемой системы, принято решение добавить в систему предварительный анализ кадров на наличие движения на нем. В том случае, когда движение обнаружено на последовательных изображениях, обрабатываемые кадры передаются для анализа каскадному классификатору признаков. Причиной данного решения является то, что процесс обработки кадра классификатором на порядок превосходит по затратам времени процесс обнаружения движения. При этом нет никакой необходимости осуществлять детектирование людей, в том случае если кадр не изменяется, а значит и человек не может там появиться. Детектор движения реализован с использованием функционала библиотеки OpenCV.

Главным компонентом системы, осуществляющим управление остальными элементами, является объект класса VideoRecorder. В основном

потоке программы создается дополнительный поток, для получения кадров с видеокамеры. С этой целью создается экземпляр класса, указанного выше. После создания объекта данного класса используется его метод StartWriting(). При старте данного метода производится инициализация объекта класса Reader, использующего функционал библиотеки FFMPEG.

Следующим этапом начинается непосредственно процесс обработки видеопотока с видеокамеры. Создается дополнительный поток, осуществляющий получение изображений с камеры с помощью объекта Reader и сохранение их во временный буфер, из которого основная часть объекта VideoRecorder будет получать видеоданные. В основном потоке обработки видео, с определенной периодичностью изображения извлекаются из буфера для последующей обработки. Если включена функция детектирования людей с предварительным детектированием движения, то полученная структура AVPacket декодируется в изображение в структуру AVFrame, из которой далее формируется элемент cv::Mat.

Далее это изображение передается на анализ детектору движения. Если на последовательных трех кадрах обнаруживается движение, то изображение передается каскадному классификатору, предварительному обученному и загруженному из файла cascade.xml. Данный классификатор обрабатывает изображение, передаваемое ему в качестве параметра в метод detectMultiScale() [5]. Результатом работы классификатора является решение о том, есть ли на изображении люди, и если есть, то по каким координатам они располагаются.

Далее эти координаты сохраняются во временный контейнер. При работе над следующим кадром, к нему применяется функция библиотеки OpenCV для межкадрового сравнения оптического Sparse-потока calcOpticalFlowPyrLK(). Данная функция принимает на вход два кадра и координаты обнаруженных объектов с первого кадра, сохраненные в массиве. Указанная функция вычисляет оптический поток для заданного набора точек, используя пирамидальный метод Лукаса-Канаде. В результате работы данная функция генерирует набор координат, описывающих положение объектов, найденных на первом кадре, на втором кадре. В случае пересечения человеком некоторой линии, выбранной за границу (например, середину кадра), программа определяет, что произошло событие «прохождение человека». При этом генерируется соответствующее уведомление для оператора, а также увеличивается счетчик прошедших людей. С определенной периодичностью в лог-файл записывается информация о текущем количестве подсчитанных людей.

В главной программе можно параллельно обрабатывать данные с нескольких камер в высоком разрешении. При тестировании велась обработка данных с трех камер, передающих видео в формате 4к. Такая производительность достигается за счет использования оптимальных

алгоритмов обработки видео. На рисунке ниже приведена структура программы (рис. 2).

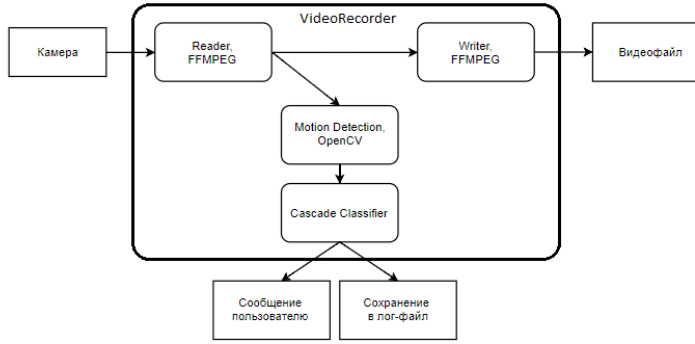

Рис. 2. Структура разработанной системы

Для разработки программного продукта изучено такое средство разработки программного обеспечения как OpenCV. Рассмотрены различные алгоритмы и структуры, используемые при работе с видео. В целях оптимизации работы системы изучен фреймворк FFMPEG позволивший достичь высокой производительности разработанной системы. В качестве интеллектуальной системы, ведущей распознавание людей выбран каскадный классификатор признаков на основе локальных бинарных шаблонов.

На основе полученных знаний разработана программа, параллельно ведущая обработку данных с нескольких камер и детектирующая людей на видео.

Литература:

1. OpenCV Library [Электронный ресурс]. 28.01.2016. - URL: http://opencv.org/ (дата доступа 24.01.2020).

2. Обзор дескрипторов изображения Local Binary Patterns (LBP) и их вариаций [Электронный ресурс]. 21.05.2016. URL : https://habr.com/post/280888

3. Обучение каскадного классификатора. [Электронный ресурс]. 14.10.2016. URL : http://recog.ru/blog/opencv/85.

4. FFMPEG [Электронный ресурс]. 11.02.2015. URL: https://www.ffmpeg.org/

5. Cascade Classifier. OpenCV. [Электронный ресурс]. 02.03.2016. URL: https://docs.opencv.org/3.0-beta/doc/tutorials/objdetect/cascade_classifier/cascade_classifier.html.