REPORTS OF NATIONAL ACADEMY OF SCIENCES OF THE REPUBLIC OF KAZAKHSTAN
ISSN 2224-5227
Volume 5, Number 5 (2014), 34 – 40

REAL TIME IMAGE RECOGNITION SYSTEM

Y.N.Amirgaliyev, M.K.Zhaparov, A.S. Sagandykova, A.K. Zhakenov
amir_ed@mail.ru
Suleyman Demirel University , Almaty , Republic of Kazakhstan

Keywords: object recognition, image processing, open programming bibliotheca
Abstract. This paper presents a mobile-oriented program for object recognition using algorithms of image processing. In particular, were analyzed results of image processing in traditional ways, described in early works, and results of parallel recognition of several objects processed in real time.

СИСТЕМА ДЛЯ РАСПОЗНАВАНИЯ ИЗОБРАЖЕНИЙ В РЕАЛЬНОМ ВРЕМЕНИ

Е.Н. Амиргалиев, М. К. Жапаров, А.С. Сагандыкова, А. К. Жакенов
amir_ed@mail.ru
Университет им. С. Демиреля, Алматы, Республика Казахстан

Ключевые слова: распознавание образов, обработка изображений, открытие программные библиотеки.
Аннотация. В данной работе рассматривается мобильно-ориентированная технология для обработки изображений. В частности, проанализированы результаты распознавания дорожных знаков как традиционными способами, приведенными в ранних работах, так и при параллельной обработке нескольких знаков в реальном времени, предложенной в данной работе.

Введение

Автотранспорт играет большую роль в развитии экономики страны. В отличие от других транспортных средств он отличается эффективностью и мобильностью. Благодаря этому объем перевозок грузов и пассажиров автомобилями растет быстрее, чем на других видах транспорта. Ежегодный прирост автомобилей на дорогах составляет 14,1%. Всего по состоянию на 1 сентября 2014 года в стране насчитывается 3 млн. 493 тыс. 922 легковых автомобиля.

Однако, несмотря на положительные факторы, влияющие на развитие экономики, существуют ряд негативных факторов, связанных с процессом автомобилизации. Сюда относятся загрязнение окружающей среды, градостроительные проблемы, связанные с выделением пространств для обеспечения движения транспортных средств, рост дефицита нефтепродуктов и т.д. К разряду наиболее отрицательных факторов процесса автомобилизации относятся ДТП и их последствия, характеризующиеся гибелью и ранением людей, материальным ущербом от повреждения транспортных средств, грузов, дорожных и иных сооружений, а также отрицательное влияние на экологию. По данным статистики за 7 месяцев текущего года произошло 10 727 ДТП, погибли 1 446 человек и 13 958 получили травмы. По оценке Всемирного Банка, ежегодно казахстанский бюджет теряет из-за дорожных аварий около 1,5 процента ВВП. К основным причинам ДТП относят рост числа автомобилей (за последние 5 лет автопарк Казахстана вырос более чем в 2 раза - с 2 до 4,2 миллиона единиц), ухудшение состояния дорог (ремонт и благоустройство дорог проводятся не столь быстрыми темпами), несоблюдение ПДД (95% всех аварий происходит из-за превышения водителем скорости или затрудненной реакции при совершении маневров) и т.д.

Из вышесказанного следует, что решение проблемы повышения безопасности на дорогах имеет большую социальную и экономическую значимость и является одной из кардинальных проблем

автомобилизации. В условиях непрерывного повышения интенсивности дорожного движения с вовлечением больших масс людей, транспортных и материальных средств деятельность по предупреждению ДТП является многоплановой и взаимосвязанной, требующей научного комплексного подхода. Одним из процессов для составления такого рода систем является процесс распознавания явлений окружающего мира.

В данной работе рассматривается система, позволяющая распознавать дорожные знаки в реальном времени. Как инструмент для разработки системы были исследованы готовые решения и предложены свои пути решения данной проблемы.

Теория распознавания образа - раздел науки, развивающий основы и методов классификации и идентификации предметов, явлений, процессов, сигналов, ситуаций и других объектов, характеризующихся конечным набором некоторых свойств и признаков. Распознавание образов как научная дисциплина начала формироваться одновременно с бывшей СССР и США примерно со второй половины 50-х годов. Построение сложных систем распознавания требует решения ряда теоретических и инженерных задач: разбиения множества объектов на классы (составление алфавита классов); выбора в условиях ограничений пространства признаков и описания на языке признаков классов объектов либо путем непосредственной обработки исходной априорной информации, либо на основе методов и алгоритмов обработки информации технических средств, а также методов и алгоритмов собственного решения задачи распознавания; разработки методов и алгоритмов оптимизации процессов распознавания в системе; оценки эффективности системы распознавания в различных режимах ее функционирования и т.д. [1]. Также для решения задач распознавания применяются библиотеки с готовыми методами и одной из таких библиотек является OpenCV.

OpenCV- библиотека компьютерного зрения с открытым исходным кодом (Open Source Computer Vision Library), разработанная лабораторией Intel в России [2]. OpenCV библиотека включает в себя библиотеки языков C и C++, служащие для разработки алгоритмов обработки изображений и машинного зрения. OpenCV совместима с другой библиотекой обработки изображений, также разработанной лабораторией Intel - Intel Image Processing Library (IPL), которая обрабатывает цифровые изображения на низком уровне, в то время как OpenCV использует алгоритмы для более точного анализа изображения [3], а также алгоритмы для определения и прослеживания объекта, анализа движений объекта, 3D реконструкции объекта и др. Благодаря простоте и эффективности методов данной библиотеки, OpenCV играет важную роль в области распознавания и анализа изображений за рубежом. По сравнению с другими готовыми решениями OpenCV имеет много преимуществ:

- написана на C и хорошо оптимизирована для использования в системах реального времени;
- совместима с другими языками программирования и платформами EiC, Ch., MATLAB;
- на данный момент является самой распространенной открытой библиотекой для распознавания объектов.

Признаки Хаара (Haar-Like features [4] предстваляют собой двоичную аппроксимацию вейвлета Хаара. Каждый признак представляет собой двоичную маску, т.е. черно-белое изображение, как можно видеть на рисунке 1.

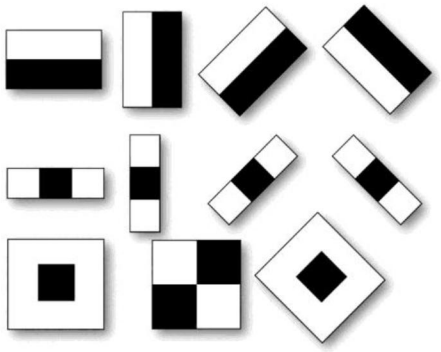

Рисунок 1- Признаки Хаара

Для описания объекта с достаточной точностью необходимо большое число признаков. Поэтому признаки Хаара должны быть организованы в каскадный классификатор.

Работа с каскадным классификатором включает два этапа: обучение и детектирование. В OpenCV есть два приложения для тренировки каскадов: opencv_haartraining и opencv_traincascade. Главное различие между этими двумя приложениями является то, что opencv_traincascade поддерживает как Хаара [5], так и LPB [6]. особенности. LPB алгоритм выполняет обучение и детектировние в несколько раз быстрее Хаара. Что касается достоверности распознавания, то все зависит от обучающей выборки. opencv_traincascade и opencv_haartraining хранят обученные классификаторы в различных форматах. Новый интерфейс обнаружения (CascadeClassifier класса в objdetect модуле) поддерживают оба формата. opencv_traincascade может сохранять обучение каскада в старом формате. Но opencv_traincascade и opencv_haartraining не может загрузить классификатор в другом формате для дальнейшего обучения после перерыва. opencv_traincascade приложение может использовать TBB для многопоточности. Чтобы использовать его в режиме многоядерных процессоров OpenCV должно быть откомпилировано с TBB.

Другие утилиты, используемые для обучения:

1. opencv_createsamples используется для подготовки учебного набора положительных и тестовых образцов в формате, который поддерживается как opencv_haartraining так и opencv_traincascade приложениями. На выходе получается файл с расширением *.vec, это двоичный формат, который содержит изображение.

2. opencv_performance может быть использовано для оценки качества классификаторов, но только для обученных opencv_haartraining. Она использует коллекцию размеченных изображений, запускает классификатор и сообщает так называемую производительность, т.е. количество найденных объектов, количество пропущенных объектов, количество ложных срабатываний и другую информацию.

Для обучения необходимо собрать образы. Есть два типа образов: негативные и позитивные. Негативные образы соответствуют отсутствию объекта на изображении (Рисунок 2). Положительные образы соответствуют изображением с обнаруженными объектами (Рисунок 3). Набор негативных образов должен быть подготовлен вручную, в то время как множество положительных образов создается с помощью утилиты opencv_createsamples.

Рисунок 2- Негативный образ

Рисунок 3-Позитивный образ.

Негативные образы должны быть взяты из произвольного изображения. Эти изображения не должны содержать обнаруженных объектов.

1. Использование алгоритмов OpenCV в реализации программы распознавания дорожных знаков

Реализация программы распознавания дорожных знаков осуществляется в два этапа: обучение каскадных классификаторов и детектирование дорожных знаков с помощью Андроид приложения. Для генерирования положительных образов применяются готовы методы библиотеки OpenCV. Результатом обучения классификатора являеься .xml файл, который в дальнейшем используется для детектирования дорожных знаков. Программа посылает видео фреймы для обработки и распознавания. В случае обнаружения знак выделяется зеленым квадратом. Блок-схема данной программы представлна ниже на рисунке 4

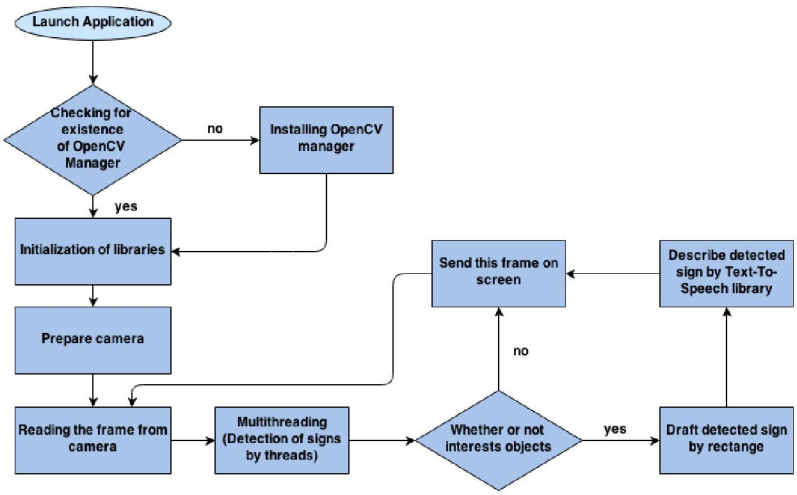

Рисунок 4- Схема работы программы

A. Обучение каскадного классификатора

Для генерирования .xml файла необходимо выбрать изображение распознаваемого объкта, как представлено на рисунке 5.

Рисунок 5- Знак пешеходного перехода

Негативные образы перечислены в специальном файле, в котором каждая строка содержит имя файла изображения (относительно каталога файла) с негативным образом.

Негативные примеры упаковываются в файл neg.dat. Для достижения более точного результата распознавания было обработано около 3000 фотографий.

Следующим этапом обучения является генерация позитивных образов. Для этого был использовани метод opencv_createsamples, который берет neg.dat как входные данные. Код программы для обучения каскадного классификатора (Рисунок 6):

```
C:\opencv\build\common\x86\opencv_traincascade.exe -data
C:\opencv\build\common\x86\lpbpeshehod -vec
C:\opencv\build\common\x86\positive.vect -bg
C:\opencv\build\common\x86\neg.dat -numPos 1000 -numNeg
3000 -numStages 20 -precalcValBufSize 512 -precalcIdxBufSize
512 -featureType LBP -minHitRate 0.999 -maxFalseAlarmRate
0.5.
```

Рисунок 6- Программный код обучения каскадного классификатора

В данном коде, “data” – файл, содержащий каскадный классификатор, “vec” – файл, содержащий информацию о позитивных образцах, “bg” – файл, содержащий информацию о негативных образцах, “numPos”- количество позитивных образцов, “numNeg”- количество негативных образцов. При генерации файла можно указывать тип каскада, в данном случае мы выбираем LPB. Как результат мы получаем файл “cascade.xml”.

B. Распознавание одного знака

OpenCV библиотека предлагает готовые интерфэйсы, методы и функции для распознавания объектов. Интерфэйсы “CVCameraViewListener2” и “VideoCapture” получают фреймы с камеры и позволяют их обрабатывать. При анализе программы с использованием интерфэйса “CVCameraViewListener2” и “VideoCapture”, скорость программы значительно повышается. Поэтому мы используем второй вариант. Инициализация объектов класса “CascadeClassifier” подгружает “cascade.xml” файл и для определения объекта на входном изображении используем метод detectMultiScale(). Метод возвращает массив, содержащий координаты интересующих объектов и передаваемый в программу для выделения объекта зеленым квадратом, как показано на рисунке 7.

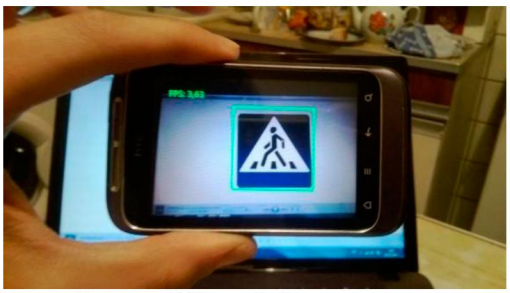

Рисунок 7-Интерфейс программы.

В зависимости от системных требований мобильного устройства программа занимает разное количество времени на распознавание одного знака. Проблема заключается в том , что с увеличением количества объектов скорость программы резко снижается. Для решения мы распараллеливаем программу с помощью нитей.

C. Принцип работы LPB

LBP представляет собой описание окрестности пикселя изображения в двоичной форме. Оператор (LBP), который применяется к пикселю изображения, использует восемь пикселей окрестности, принимая центральный пиксель в качестве порога. Пиксели, которые имеют значения больше, чем центральный пиксель (или равное ему), принимают значения "1", те, которые меньше центрального, принимают значения "0". Таким образом получается восьмиразрядный бинарный код, который описывает окрестность пикселя (Рисунок 8).

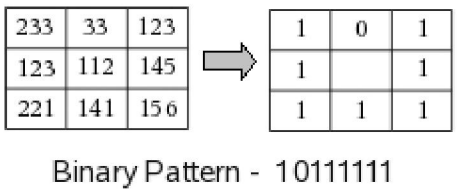

Рисунок 8-Описание работы LBP

В десятичной форме результирующее восьмиричное слово (LBP код) может быть представлено в следующем виде:

$$LBP(x_c, y_c) = \sum_{n=0}^{7} s(i_n - i_c)2^n \quad ,$$

где i_c – соответствует центральному пикселю () in – относится к восьми окружающим пикселям. Функция s(x) описывается как

$$s(x) = \begin{cases} 1 & \text{if} \quad x \geq 0 \\ 0 & \text{if} \quad x < 0 \end{cases}.$$

Каждый бит кода LBP имеет тот же уровень значимости и, что два последовательных значения бит. Код LBP можно интерпретировать как индекс структуры ядра. По определению, оператор LBP не зависит от какого-либо монотонного серо-масштабного преобразования, сохраняющего пиксели порядка интенсивности в местные окрестности (Рисунок 9).

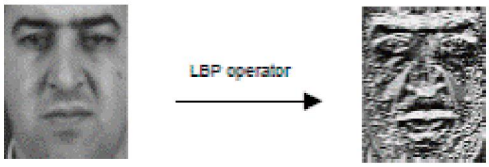

Рисунок 9- Результат работы оператора LBP (слева исходное изображение, справа обработанное изображение)

Благодаря своей текстуре LBP является не сложным в реализации и являетсяс популярным в распознавании образов. В последнее время LBP была применена в работах для обнаружения лиц, распознавания лиц, поиска изображений, обнаружения движений и визуального осмотра[7].

D. Распознавание нескольких знаков

Распознавание одного знака менее эффективно, так как на дорогах одновременно всьречаются несколько знаков. Для решения этой проблемы мы распараллеливаем программу, создавая нити для каждого знака. Каждая нить распознает определенный вид знака, таким образом сокращая время работы программы в несколько раз. Главный метод описан ниже (Рисунок 10).

В методе frame – текущий фрэйм, mJavaDetector()- главный метод распознавания, который использует файл cascade.xml.

Для озвучивания ниименования знака используется метод speak() библиотеки Android Text-To-Speech().

Заключение

Данная программа была реализована, используя основые методы библиотеки OpenCV интегрированную с мобильной платформой Android. Все используемы методы были в результате анализа всех имеющихся методов и подбором более оптимального варианта, учитывая особенности данной системы. Преимущества программы:

- Релизация на мобильной платформе
- Распознавание знаков в реальном времени
- Распознавание нескольких знаков
- Озвучивание наименования знака

Данная программа планируется использоваться в более больших системах, использующих концепцию “Smart Auto”.

```
Private void detectObject(CascadeClassifier mJavaDetector, Mat frame){
        Mat frameGray = new Mat();
        if(doDetect){
                 // Convert the RGB frame to HSV as it is a more appropriate format when calling Core.inRange
                 Imgproc.cvtColor(frame,frameGray,Imgproc.COLOR_BGR2GRAY);
                if (mAbsoluteFaceSize == 0) {
                  int height = frameGray.rows();
                  if (Math.round(height * mRelativeFaceSize) > 0) {
                       mAbsoluteFaceSize = Math.round(height * mRelativeFaceSize);
                  }
                 }
                //detect objects which will be recognized
                 MatOfRect signs = new MatOfRect
                 if (mJavaDetector != null){
                       mJavaDetector.detectMultiScale(frameGray, signs, 1.1, 2, 2, new Size(mAbsoluteFaceSize,
mAbsoluteFaceSize),
                      new Size());
                 }
                 //set rectangles to our frame
                 Rect[] signsArray = signs.toArray();
                 for (int j = 0; j < signsArray.length; j++){
                      signsGlobalList.add(signsArray[j]);
                     //Core.rectangle(mCurrentFrame, signsArray[j].tl(), signsArray[j].br(), SIGN_RECT_COLOR, 3);
                 }
                 if(!tts.isSpeaking() && signsArray.length>0){
                       tts.speak(signToTextMap.get(mJavaDetector),      TextToSpeech.QUEUE_FLUSH, null);
                 }
           }}
```

Рисунок 10-Обработка нескольких знаков

REFERENCES

[1] Е. Н. Амиргалиев Теория распознавания образов и кластерного анализа. КазНТУ, 2003, 360 с.

[2] Song, Li, Xijian. Ping, Yihong. Ding, "Application of open source computer vision library," Computer Applications and Software, vol. 8, pp. 134-136, August 2005

[3] Open Source Computer Image Library Reference Mannual. 2001(8)

[4] Jones, M., Viola, P. (2001) Robust Real-Time Face Detection. URL: International Journal of Computer Vision, 57(2), 137-154, 2004. Available from: http://www.vision.caltech. edu/html-files/EE148-2005-Spring/pprs/viola04ijcv.pdf (ссылка проверена 28 мая 2013)

[5] Paul Viola, Michael Jones. Rapid Object Detection using a Boosted Cascade of Simple Features. Conference on Computer Vision and Pattern Recognition (CVPR), 2001, pp. 511-518

[6] Shengcai Liao, Xiangxin Zhu, Zhen Lei, Lun Zhang and Stan Z. Li. Learning Multi-scale Block Local Binary Patterns for Face Recognition. International Conference on Biometrics (ICB), 2007, pp. 828-837

[7] Sebastien Marcel, Yann Rodriguez and Guillaume Heusch, "On the Recent Use of Local Binary Patterns for Face Recogntion", International Journal of Image and Video Processing, Special Issue on Facial Image Processing, May 25, 2007

Резюме

Е.Н. Әмірғалиев, М.Қ. Жапаров,Ә.Н. Сағындықова, А.Қ. Жакенов

([1]Сүлейман Демирель атындағы университет, Алматы, Қазақстан Республикасы)

КЕСКІНДЕРДІ НАҚТЫ УАҚЫТТА ТАНУ ЖҮЙЕЛЕРІ

Түйін сөздер: бейне тану, кескін өңдеу, ашық бағдарламалық кітапхана

Мақалада ұялы телефон арқылы кескінді өңдеуге арналған бағдарлама қарастырылған. Атап айтқанда, жол белгілерін тану әдісі ұсынылған. Кескін танудың бұрынғы жұмыстарда көрсетілген әдістеріді қолданумен қатар осы жұмыста ұсынылған бірнеше белгілерді бір уақытта параллель өңдеу және видео режимде тану нәтижелері көрсетілген.

Поступила 21.11.2014 г.