ПАРАЛЛЕЛЬНЫЕ АЛГОРИТМЫ РЕШЕНИЯ ЗАДАЧИ ОПРЕДЕЛЕНИЯ ДВИЖЕНИЯ НА ОСНОВЕ ВИДЕОИНФОРМАЦИИ

Щербаков М.С.[1], Борисов А.П.[2] Email: Shcherbakov666@scientifictext.ru

[1]Щербаков Михаил Сергеевич – магистрант,
кафедра прикладной математики;
[2]Борисов Алексей Павлович – кандидат технических наук, доцент,
кафедра информатики, вычислительной техники и информационной безопасности,
Алтайский государственный технический университет им. И.И. Ползунова,
г. Барнаул

Аннотация: *в данной статье анализируется задача поиска областей движения в потоке изображений. Рассмотрена реализация поиска движения на видео с помощью методов библиотеки EmguCV. Разработан алгоритм работы с библиотекой в многопоточном режиме. Получены результаты тестирования программного обеспечения на различных исходных данных. Эксперимент показал, что скорость обработки данных увеличивается вне зависимости от разрешения изображений. Разработанная реализация алгоритма поиска областей движения подходит для анализа присутствия движения в видеопотоке, но не определения его четких границ.*
Ключевые слова: *компьютерное зрение, поиск движения, EmguCV, многопоточные алгоритмы.*

PARALLEL ALGORITHMS OF THE SOLUTION OF THE PROBLEM OF DEFINITION OF THE MOVEMENT ON THE BASIS OF THE VIDEO INFORMATION

Shcherbakov M.S.[1], Borisov A.P.[2]

[1]Shcherbakov Mikhail Sergeyevich - Undergraduate,
DEPARTMENT OF APPLIED MATHEMATICS;
[2]Borisov Alexey Pavlovich - candidate of technical sciences, associate professor,
DEPARTMENT OF INFORMATICS, COMPUTER FACILITIES AND INFORMATION SECURITY,
ALTAI STATE TECHNICAL UNIVERSITY OF I.I. POLZUNOV,
BARNAUL

Abstract: *in this article the problem of search of areas of the movement in a flow of images is analyzed. Implementation of search of the movement on video by means of methods of EmguCV library is considered. The operation algorithm with library in the multithreaded mode is developed. Results of software testing on different basic data are received. The experiment showed that data processing rate increases regardless of permission of images. The developed implementation of the search algorithm of areas of the movement is suitable for the analysis of presence of the movement at a video flow, but not definition of its clear boundary.*
Keywords: *computer sight, search of the movement, EmguCV, multiline algorithms.*

УДК 004.021

В настоящее время компьютерное зрение приобретает всё большую популярность и является одной из востребованных и быстроразвивающихся областей. Имеется очень широкий спектр задач, решаемых технологиями компьютерного зрения, одной из которых, является определение движения объектов. Сегодня оно используется повсеместно. Этот факт делает работу в области оптимизации алгоритмов данной задачи весьма актуальной.

В данной статье задача определения движения (областей движения) рассматривается в условиях неподвижной камеры и относительно постоянного фона. Подобные задачи, как правило, возникают при разработке охранных систем слежения. Целью работы является увеличение скорости обработки видеопотока, за счет решения задачи обнаружения областей движения в кадре в параллельном режиме.

Задача выделения областей движения на видео – одна из классических задач компьютерного зрения. Входом указанной задачи является последовательность кадров некоторого I1, I2, … IN видео [1]:

$$I_k = \{I_k(x,y), 0 \le x < width, 0 \le y < height\}, k = \overline{1,N}, \quad (1)$$

где: width – ширина кадра, height – высота кадра, а Ik(x, y) в общем случае представляет собой вектор фиксированной размерности.

Решением настоящей задачи является совокупность областей изображения для каждого кадра видео, в которых происходит движение одного или нескольких объектов. Для решения поставленной задачи в работе используется метод вычитания фона. Модель фона строится при помощи «Смеси Гауссовых распределений».

Приложение представляет собой одно окно «WindowsForm», которое позволяет открыть два потока видео с Веб камеры, или видеофайла, запустить процесс отслеживания движений в них и в режиме реального времени. Видеопоток создается клиентским приложением, с помощью метода библиотеки EmguCV VideoCapture() либо GetCaptureProperty(), в зависимости от того, какой источник изображения (камера или видеофайл соответственно) был выбран [2].

После выбора всех желаемых видеопотоков из клиентского приложения происходит запуск обработки путем вызова метода Start каждого объекта VideoCapture. При этом в каждом VideoCapture запускается отдельный поток, выполняющий покадровую обработку видеопотока.

Для обработки из видеопотока каждый раз извлекается следующий кадр и обрабатывается в методе NewProcessFrame следующим образом. С помощью метода Retrieve, библиотеки EmguCV, полученный кадр преобразовывается в оттенки серого. С помощью, имеющейся в EmguCV, имплементации алгоритма выделения фона, получается маска кадра, отражающая объекты переднего плана на кадре. На основе полученной маски кадра, вычисляются области движения. Каждая область движения записывается в отдельный объект типа Rectangle. На основном кадре область выделяется прямоугольников, диагональ которого равна расстоянию между крайними точками области движения. Кадр, с выделенными областями движения выводиться на ImageBox. После окончания работы метода NewProcessFrame, замеряется время с начала работа метода, до его завершения, и выводится в listBoxTime. Так же пересчитываются среднее и максимальное время обработки кадров.

В процессе выполнения работы был разработан алгоритм с ускоренной обработкой изображений. Так как основные вычисления для поиска областей движения делаются с помощью методов библиотеки EmguCV, было принято решение оптимизировать области кода, предшествующие каждому этапу обработки кадра. Оптимизация алгоритма заключается в распараллеливании обработки кадра на трех этапах, описанных ниже.

В последовательном алгоритме в один момент времени обрабатывается только один кадр, полученный из файла или веб камеры. В новой версии обработка кадров идет следующем образом. Первые 4 кадра сохраняются в 4 объекта типа Mat. После загрузки 4-го кадра начинается обработка кадров в отдельных потоках. После окончания обработки первого кадра, он выводится на интерфейс пользователя. В освободившийся поток записывается следующий кадр, в это же время выводится кадр из второго потока, в нём начинается обработка следующего кадра и так далее. Данное решение позволяет уменьшить среднее время обработки каждого кадра.

Следующим этапом распараллеливания алгоритма стала работа метода обработки кадров с частью изображения (весь кадр / кол-во потоков).

Перед подачей передачей массива, содержащего текущий кадр для обработки в метод ProcessFrame, массив разбивается на равные области, каждая из которых обрабатывается в отдельном потоке. После завершения обработки каждой части массива, в исходный массив записываются обновленные данные, после чего выводятся на интерфейс пользователя. Плюсы такого решения: средняя скорость обработки каждого кадра на 4 потоках выросла в среднем в 2,1 раза. В тоже время, данное решение имеет серьезный недостаток. Области одного движения, расположенные в частях кадра, обрабатываемых в разных потоках, обозначаются как отдельные, что является некомфортным для пользователя. Существуют пути исправления данного недостатка, но в работе реализованы не были.

Третьим этапом доработки стала обработка всех найденных областей движения в нескольких потоках. Для организации параллельных вычислений используются класс Threading, содержащий метод AsParrallel. Внутри метода осуществляется разбиение цикла на части по количеству потоков (настраивается в этом классе константой THREADS_COUNT или определяется автоматически) и запуск каждой части в отдельном потоке фиксированного пула потоков, после чего ожидается завершение выполнения всех подзадач.

После реализации программы проводились эксперименты с замерами скорости работы алгоритма в обработке видео в режиме реального времени. Замеры выполнялись на процессоре Intel Core i5 4460K (4 ядра, 8 потоков). Тестирование проводилось путем измерения среднего времени обработки всех полученных кадров, и сравнения результатов работы последовательного и параллельного алгоритмов. Результаты проведенных замеров приставлены в таблице 1.

Эксперимент показал, что распараллеливание данного алгоритма целесообразно на видео с любыми разрешениями. В то же время при обработке видео с большей изменения кадра, наблюдается уменьшение скорости вывода кадров на экран. Стоит отметить, что данная реализация детектирования движения на видео с одной стороны ускоряет работу алгоритма, но с другой искажает плавность перехода кадров и корректность отображения областей движений.

Таблица 1. Результаты проведенных экспериментов

Источник видео	**Разрешение (пикс.)**	**Время (секунды)**	**Используемый алгоритм**	
			последовательный	**параллельный**

Веб камера	640 x 480	60	23,1 мс	18,1 мс
Веб камера	800 x 600	60	27,1 мс	20,3 мс
Веб камера	1280 x 720	60	32,3 мс	25,3 мс
Видеофайл	500 x 276	8	32,77 мс	23,4 мс
Видеофайл	728 x 408	24	45, 3 мс	34,77 мс
Видеофайл	474 x 360	324	35,4 мс	26,1 мс

В рамках проделанной работы реализовано программное обеспечение, выполняющие поиск областей движение методом вычитания фона, с применением параллельных вычислений. Разработанный вариант реализации алгоритма детектирования изображения может найти своё применение в системах, где необходимо в режиме реального времени определить факт движения в контролируемой области, без идентификации его четких границ.

Список литературы / References

1. *Кустикова В.Д.* Учебный курс «Разработка мультимедийных приложений с использованием библиотек OpenCV и IPP». НГУ им. Лобачевского, 2013. Нижний Новгород. 34 с.
2. EmguCV. Tutorial. [Электронный ресурс]. Режим доступа: http://www.emgu.com/wiki/index.php/Tutorial Загл. с экрана/ (дата обращения: 18.06.2019).
3. *Bradski G., Kaehler A.* Learning OpenCV Computer Vision with OpenCV Library. O' Reilly Media Publishers, 2008. 571 p.