Сентябрь, № 6 **Физико-математические науки** **2014**

УДК 004.932

АЛЕКСЕЙ БОРИСОВИЧ СЕМЕНЦОВ
директор Центра коллективного пользования научным оборудованием, Петрозаводский государственный университет (Петрозаводск, Российская Федерация)
alsementsov@gmail.com

АЛЕКСАНДР НИКОЛАЕВИЧ ВОРОБЬЕВ
аспирант кафедры информационно-измерительных систем и физической электроники физико-технического факультета, Петрозаводский государственный университет (Петрозаводск, Российская Федерация)
alexn.vorobyev@gmail.com

МАРИЯ АЛЕКСАНДРОВНА СЕРЕЖИНА
аспирант кафедры информационно-измерительных систем и физической электроники физико-технического факультета, Петрозаводский государственный университет (Петрозаводск, Российская Федерация)
serejinam@gmail.com

АЛГОРИТМ ПОИСКА СВЯЗНЫХ ОБЛАСТЕЙ НА БИНАРНОМ ВИДЕОИЗОБРАЖЕНИИ ЧЕРЕССТРОЧНОЙ РАЗВЕРТКИ ДЛЯ ПРИМЕНЕНИЯ В FPGA*

Представлена модификация аппаратно ориентированных алгоритмов поиска связных областей на бинарном видеоизображении, предназначенная для чересстрочной развертки. Предложен вариант сканирующего окна для чересстрочной развертки и алгоритм его конвейерной обработки в FPGA, позволяющий проводить вычисления в потоковом режиме с частотой кадров 25 Гц без их буферизации. Применение других известных алгоритмов потребует буферизации кадров, так как они ориентированы на прогрессивную развертку.

Ключевые слова: видеоизображение с чересстрочной разверткой, поиск связных областей, однопроходной алгоритм, сканирующее окно, FPGA

ВВЕДЕНИЕ

В данной статье представлено описание аппаратно ориентированного алгоритма поиска связных областей бинарной маски и описывающих их прямоугольников на видеоизображении. Под связной областью понимается область изображения, где пиксели примыкают друг к другу (соседствуют) и при этом имеют некоторые схожие свойства, например цвет.

Задача решалась в рамках проекта по созданию «Устройства обнаружения пламени на видеоизображении» (использовался стандарт PAL 720x576, 25 Гц). В ходе проекта была разработана программная модель обнаружения пламени [1] в среде Simulink (Matlab), для обеспечения работы в режиме реального времени алгоритмы переведены на программируемые логические интегральные схемы FPGA (англ. field-programmable gate array).

Суть алгоритма поиска пламени сводится к поиску пикселей, имеющих цвет, сходный по цветовой гамме с пламенем, выделению таких пикселей в восьмисвязные прямоугольные области, объединению пересекающихся областей в области интереса с последующим вычислением и анализом для них динамических характеристик. Объединение пересекающихся связных областей позволяет надежно объединять языки одного пламени в единое целое.

В среде Simulink для поиска связных областей и описывающих эти области прямоугольников использовалась встроенная функция «Blob Analysis», работающая с записанным в оперативную память кадром.

Разрабатываемое устройство должно работать с потоковыми видеоданными чересстрочной развертки без буферизации и пропуска кадров. Функция «Blob Analysis» не соответствует этим требованиям и не может быть использована.

Обзор существующих методов выделения связных областей, ориентированных на работу в FPGA, показал, что поиск в них осуществляется либо по уже записанному кадру, что требует больших объемов оперативной памяти для задержки видео [3], [5], либо в потоковом видеосигнале, но для прогрессивной развертки [2], [4]. Стоит отметить, что основной задачей указанных алгоритмов является нумерация бинарной маски и составление таблицы связности, которую требуется далее анализировать; практически отсутствует информация о нахождении координат прямоугольников, описывающих связные области, что является основной задачей разрабатываемого алгоритма.

В итоге было принято решение разработать собственный однопроходной алгоритм составления предварительной таблицы координат связных областей. При этом постараться достичь

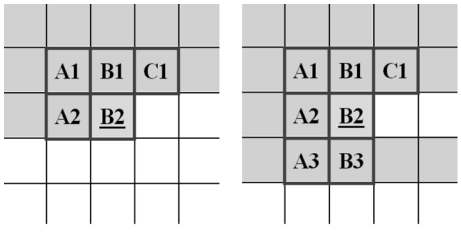
минимального количества предварительно найденных связных областей, так как это существенно влияет на время работы последующего блока поиска пересечений, который должен успевать обрабатывать данные менее чем за кадр.

МЕТОД ВЫДЕЛЕНИЯ СВЯЗНЫХ ОБЛАСТЕЙ

Аппаратный поиск восьмисвязных пикселей в основном решается использованием одно- и двухпроходных алгоритмов, а также метода «выжигания области» (описаны в [4]).

Двухпроходной алгоритм потребует буферизации в памяти результатов первого прохода для двух кадров (объем не менее 10 Мбит), а также приведет к необходимости буферизировать исходные кадры и все данные для дальнейшего анализа. Метод «выжигания области» также требует буферизации кадров: исходного и хранящего промежуточный результат. Использование дополнительных банков внешней памяти в проекте нецелесообразно, поэтому два этих метода не рассматривались.

В существующих однопроходных алгоритмах для прогрессивной развертки [2], [4] используется сканирующее окно (рис. 1а), в котором текущая ячейка «B2» проходит последовательно каждый пиксель кадра. В ходе прохода формируется таблица связности, которая в процессе прохода и по его окончании анализируется.

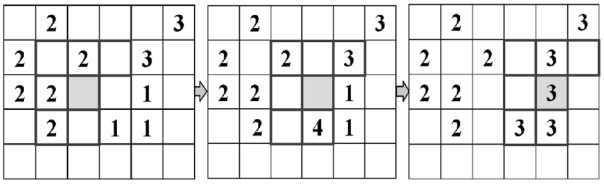


а б

Рис. 1. Сканирующие окна

В чересстрочной развертке вначале выводятся нечетные строки кадра (1, 3, …), а затем четные (2, 4, …). Маркировка метками (нумерация) пикселей в каждом поле (как в [2]) приведет к резкому росту количества используемых меток при проходе первого поля, так как будет отсутствовать информация о пикселях второго поля. Это увеличит размерность таблицы координат связных областей в несколько раз, что приведет к многократному росту времени поиска в ней возможных пересечений. Поэтому анализ окружения для текущего пикселя и составление таблицы координат областей осуществляются только по приходящим пикселям четных строк, когда во временную память уже записаны нечетные строки кадра.

На рис. 1б показан выбранный для анализа вариант сканирующего окна. B2 является текущим бинарным пикселем четного поля, A2 – предыдущим. A1, B1, C1 – значения меток для пикселей верхней нечетной строки. A3, B3 – бинарные пиксели нижней нечетной строки. На каждом шаге происходит перезапись единичных значений пикселей меткой с минимальным номером среди соседей, если соседи присутствуют (рис. 2а, в). В случае отсутствия соседства – новой меткой с более высоким по очереди номером (рис. 2б), для чего предусмотрен указатель текущего свободного номера метки.



а б в

Рис. 2. Проход сканирующего окна

На текущем пикселе из оперативной памяти требуется считать только значения C1 и B3, все остальные значения элементов берутся от предыдущего шага, а именно: A1 = B1, B1 = C1, A2 = B2, A3 = B3. Исключение составляют пиксели за границами изображения, они обнуляются.

Значение C1 считывается из буферной памяти строки, запись в которую производилась при проходе предыдущей строки четного поля промаркированными элементами A3. В случае первой четной строки данные значения равны либо 0, либо 1 и считываются из буферной памяти маски первого поля. Значение B3 считывается из буферной памяти маски первого поля.

Особенность выбранного окна в том, что используется выступ C1, метка которого уже известна из анализа предыдущей четной строки. Это позволяет уменьшить количество создаваемых меток.

Таблица координат меток обновляется только по значениям A1 и A2. Считанные из таблицы текущие значения координаты области сверяются с текущими координатами A1 и A2, если данные пиксели расширяют эту область, то координаты перезаписываются.

Использование варианта обновления только по двум пикселям A1 и A2 позволяет исключить ситуацию, изображенную на рис. 2б, когда формируется новая метка («4»), а на следующем такте она переименовывается (в «3»). Количество пикселей для метки «4» не обновляется и остается равным нулю. При последующей отправке данных таблицы в блок поиска пересечений такие нулевые элементы фильтруются. Обновление по A1, A2 позволяет считывать и обновлять на каждом шаге только один элемент в памяти таблицы, это сводит к минимуму количество обращений к памяти и тем самым уменьшает время обработки.

Когда вывод второго поля завершится, таблица будет готова для дальнейшего поиска в ней

пересекающихся областей. Она является предварительной, так как один последовательный проход не позволяет разметить одной меткой такие элементы на изображении, как, например, буква «Ш» (подробное описание таких случаев представлено в [2]). В данной работе обработка таких ситуаций происходит совместно с поиском пересечений областей, это позволяет объединить области, которые при проходе не были объединены, но являются соседями.

Разработанный на языке Verilog модуль для FPGA работает на частоте 54 МГц, что в 4 раза больше пиксельной частоты. Он производит конвейерную обработку и состоит из трех процессов (рис. 3). Первый процесс захватывает входные данные (1, 2), второй анализирует и маркирует пиксели в окне (3, 4), а третий обновляет координаты области в таблице (5, 6, 7).

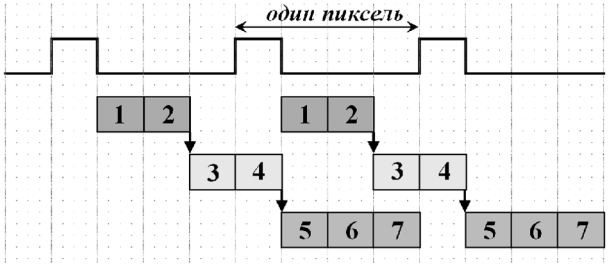


Рис. 3. Конвейерная обработка

1. Захват координат и маски. Операции A1 = B1, A2 = B2, B3 = B3, B1 = C1;
2. Чтение из памяти C1, B3. Проверка и выполнение граничных условий;
3. Анализ меток в окне;
4. Анализ меток в окне. Выставление адреса чтения для таблицы. Запись A3;
5. Ожидание данных из таблицы;
6. Анализ расширения области и подсчет количества пикселей;
7. Обновление координат области.

Объем использованной внутренней памяти (таблица) составляет не более 5 % встроенной памяти для используемой микросхемы Altera Cyclone V 149K.

Объем задействованной памяти

Наименование памяти	Объем, Кбит
Бинарная маска первого поля	256
Буфер верхней строки	10
Таблица координат областей	60
ИТОГО	326

По результатам компиляции в среде Quartus прирост логических ячеек после внедрения описанного алгоритма в общий проект составил 4 %.

ВАРИАНТ ОКНА С 10 ПИКСЕЛЯМИ

Так как в варианте окна с 7 пикселями существует запас по времени работы процесса анализа и маркировки меток, было рассмотрено сканирующее окно из 10 пикселей (рис. 4б).

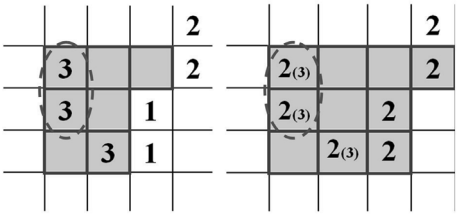


Рис. 4. Пример обработки конвейерного окна на 7 (а) и 10 (б) пикселей

На рис. 4б представлена ситуация, когда данный вариант позволяет оставить в таблице нулевым элемент № 3, в отличие от окна на рис. 4а. При реализации потребовалась частота в 7 раз больше пиксельной (94,5 МГц), так как увеличилось время анализа окна. Основные принципы: работа с таблицей по A1 и A2, а также запись в буфер строки только значения A3 – были сохранены. С учетом фильтрации элементов с нулевым количеством пикселей применение окна с 10 пикселями позволило сократить на 3–10 % количество элементов таблицы по сравнению с окном из 7 пикселей. Дальнейшее удлинение окна по горизонтали позволяет еще снизить количество элементов таблицы. Расширение окна еще до 13 пикселей возможно при несущественных изменениях в разработанном алгоритме и его аппаратной реализации. Дальнейшее расширение возможно до определенной стадии за счет дробления на более мелкие процессы. Поиск предельного значения не производился, так как окно из 10 пикселей обеспечивает требуемое быстродействие.

ЗАКЛЮЧЕНИЕ

В реализации на FPGA было отмечено улучшение точности выделения пламени по сравнению с моделью в Matlab, где для поддержания приемлемого быстродействия (до 3 кадров/с) была установлена фильтрация малых областей в Blob Analysis, обеспечившая максимальное количество возможных областей на выходе не более 50. В аппаратной реализации фильтрация по размерам была проведена после поиска пересечений, в результате – также менее 50 областей, но с быстродействием уже 25 кадров/с.

Описание применяемого метода поиска пересечений выходит за рамки данной статьи, но можно отметить, что аппаратно реализованный алгоритм поиска связных областей совместно с последующим блоком поиска пересечений обеспечил требуемое потоковое выделение областей интереса для созданного устройства обнаруже-

ния пламени. При тестировании максимальное количество областей в таблице для окна с 10 пикселями достигало 615, для большинства изображений – в среднем 120. Для таблицы, содержащей 615 областей, поиск пересечений выполнялся за время вывода 65 строк видеоизображения, тогда как обновление таблицы для следующего кадра начинается через 337 строк (первое поле и два гасящих импульса). В итоге вычисления производятся с запасом по времени.

Алгоритм реализован на семействе микросхем FPGA Altera низкого ценового диапазона без применения внешних микросхем памяти. При незначительных доработках алгоритм может быть адаптирован под более высокое разрешение.

* Работы выполнялись по заказу ЗАО «Инженерный центр пожарной робототехники ‘‘ЭФЭР’’» г. Петрозаводск.

СПИСОК ЛИТЕРАТУРЫ

1. Воробьев А. Н., Семенцов А. Б., Мощевикин А. П. Разработка программной модели распознавания огня на видеоизображении // Материалы 65-й научной конференции студентов, аспирантов и молодых ученых. Петрозаводск: Изд-во ПетрГУ, 2013. С. 166–167.
2. Bailey D. G., Johnston C. T. Single Pass Connected Components Analysis. Proceedings of Image and Vision Computing New Zealand 2007. Hamilton, New Zealand, 2007. P. 282–287.
3. Trein J., Th. Schwarzbacher A., Hoppe B. FPGA Implementation of a Single Pass Real-Time Blob Analysis Using Run Length Encoding, MPC-Workshop, Ravensburg-Weingarten, Germany, 2008. P. 71–77.
4. Walczyk R., Armitage A., Binnie T. D. Comparative Study on Connected Component Labeling Algorithms for Embedded Video Processing Systems, IPCV, CSREA Press, VOL. 2, 2010.
5. Yasuaki Ito, Koji Nakano. Low-Latency Connected Component Labeling Using an FPGA. Int. J. Found. Comput. Sci. 21, 2010. P. 405–425.

Sementsov A. B., Petrozavodsk State University (Petrozavodsk, Russian Federation)
Vorob’ev A. N., Petrozavodsk State University (Petrozavodsk, Russian Federation)
Serezhina M. A., Petrozavodsk State University (Petrozavodsk, Russian Federation)

CONNECTED COMPONENT LABELING ALGORITHM ON BINARY INTERLACED VIDEO FOR USE IN FPGA

This article presents a modification of the known hardware-oriented algorithms for finding connected components in the binary video image, designed for interlaced scanning. The proposed variant of a scanning window for interlaced scanning and an algorithm of its pipeline processing in FPGA allow carrying out calculations in a streaming mode at the frame rate of 25 Hz without buffering. The existing algorithms require frame buffering as they are intended for progressive scanning.

Key words: interlaced video, connected component labeling, single pass algorithm, scanning window, FPGA

REFERENCES

1. Vorob’ev A. N., Sementsov A. B., Moshchevikin A. P. Development of the software model of fire detection in video [Razrabotka programmnoy modely raspoznavaniya ognya na videoizobrazhenii]. *Materialy 65-y nauchnoy konferentsii studentov, aspirantov i molodykh uchenykh* [Proceedings of the 65th scientific conference of students and young scientists]. Petrozavodsk, PetrSUPubl., 2013. P. 166–167.
2. Bailey D. G., Johnston C. T. Single Pass Connected Components’ Analysis. Proceedings of Image and Vision Computing New Zealand 2007. Hamilton, New Zealand, 2007. P. 282–287.
3. Trein J., Th. Schwarzbacher A., Hoppe B. FPGA Implementation of a Single Pass Real-Time Blob Analysis Using Run Length Encoding, MPC-Workshop, Ravensburg-Weingarten, Germany, 2008. P. 71–77.
4. Walczyk R., Armitage A., Binnie T. D. Comparative Study on Connected Component Labeling Algorithms for Embedded Video Processing Systems, IPCV, CSREA Press, VOL. 2, 2010.
5. Yasuaki Ito, Koji Nakano. Low-Latency Connected Component Labeling Using an FPGA. Int. J. Found. Comput. Sci. 21, 2010. P. 405–425.

Поступила в редакцию 26.05.2014