УДК 004.021, 004.415

© 2020 г. **В.Л. Дубровин**

(Поволжский государственный технологический университет, Йошкар-Ола),

М.В. Петропавловский, д-р техн. наук

(Марийский государственный университет, Йошкар-Ола)

ТЕХНОЛОГИЯ МНОГОПОТОЧНОЙ ОБРАБОТКИ ДАННЫХ В ПРОГРАММНОМ КОМПЛЕКСЕ РАСПОЗНАВАНИЯ НОМЕРОВ ТРАНСПОРТНЫХ СРЕДСТВ «FBEHOLDER»

Представлена технология создания системы многопоточной обработки данных на основе информационного графа и конвейерной обработки в задаче распознавания номеров транспортных средств. Приведено описание архитектуры разработанного программного комплекса «FBeholder», а также структура графа разбора видеоданных.

Ключевые слова: обработка видеоданных, многопоточные вычисления, распознавание образов.

DOI: 10.22250/isu.2020.63.116-124

Введение

Задача распознавания автомобильных номеров не является новой. В ряде статей рассматриваются алгоритмы распознавания номеров с описанием этапов преобразования графического изображения и представлением блок-схем процесса распознавания [1, 2]. Отметим, что работы ориентированы на описание этапов обработки изображения и методов, которые применяются для получения из входящего кадра отфильтрованного бинарного изображения, пригодного для выделения связанных групп символов с применением соответствующих методов их распознавания. При этом никак не затрагивается вопрос создания программной архитектуры, позволяющей выполнять обработку графических данных в многопоточном режиме с максимально эффективным задействованием всего потенциала современных многоядерных процессоров, а также не рассматриваются возможности использования

графической подсистемы для параллельной обработки данных. Сам алгоритм распознавания автомобильных номеров от захвата исходного кадра до получения строки распознанного номерного знака зачастую представлен как статическая структура и имеет ограничения, поскольку не позволяет гибко адаптировать программу без модификации исходного кода как для различных условий съемки (день, ночь, наличие контрастной или градиентной засветки), так и для вычислительных систем разной мощности (например, изменением количества уровней порогового преобразования можно повышать или понижать качество распознавания, подстраивая общую скорость процесса обработки данных для конкретной аппаратной системы).

В статье рассматривается технология многопоточной обработки данных с использованием информационного графа [3] и вычислительного конвейера [4] применительно к задаче распознавания номеров транспортных средств (ТС). Данная технология положена в основу программного комплекса «FBeholder», позволяющего эффективно распознавать государственные регистрационные знаки (ГРЗ) и вычислять параметры траектории движения ТС (координаты, скорость) по видеоданным даже при выполнении на офисных компьютерах начального уровня. Предлагаемый подход может быть использован и для решения других задач, где на входе имеется некий поток данных, а на выходе посредством применения последовательности преобразований (фильтров) в параллельном многопоточном режиме извлекаются требуемые данные для представления пользователю. Разработанная программная архитектура укладывается в концепцию MVC (Model-View-Controller) [5], а сам информационный граф и используемые в нем фильтры за счет модульной организации могут быть легко изменены или дополнены по технологии plug-in.

Постановка задачи

При разработке архитектуры программного комплекса распознавания номеров были сформулированы следующие требования:

программный комплекс на уровне архитектуры и используемых алгоритмов должен быть адаптирован для эффективного выполнения в многопоточных, многоядерных, многопроцессорных системах, при этом архитектура программного комплекса должна допускать возможность использования ресурсов графической подсистемы посредством выполнения гибридных параллельных вычислений при обработке данных;

степень многопоточности при выполнении обработки данных должна быть величиной настраиваемой, зависящей от алгоритма разбора данных;

алгоритм разбора данных должен быть настраиваемым в зависимости от конкретной задачи посредством описания схемы разбора данных и модульного пред-

ставления с использованием технологии plug-in, без необходимости изменения исходного кода программного комплекса;

архитектура программного комплекса должна быть организована по схеме разделения данных приложения, пользовательского интерфейса и управляющей логики на три отдельных компонента с использованием концепции «Модель-Представление-Контроллер».

Архитектура программного комплекса

Распознавание номера ТС реализуется последовательно на следующих трех уровнях (рис. 1):

статический уровень – осуществляет захват и распознавание отдельного статического кадра;

динамический уровень – осуществляет аккумуляирование результатов распознавания статических кадров во времени с использованием алгоритмов слежения;

уровень представления – реализует получение аккумуляированного результата распознавания для отображения в приложении, сохранения в базе данных (БД) или транслирования информации приложению-клиенту.

Рис. 1. Уровни распознавания номера транспортного средства.

Статический уровень включает в себя все этапы обработки отдельного кадра – от извлечения статического изображения с камеры (или из файла) до получения

строк символов распознанных номерных пластин. В процессе распознавания статического кадра также выполняется получение дополнительной информации – такой как тип номерной пластины, достоверность распознавания и др. Вся информация о распознанной номерной пластине сохраняется и передается на следующий уровень.

Отметим, что в отличие от аналогичных программных комплексов, использующих результат распознавания статического кадра как окончательный [6], результат, полученный на данном этапе, не может рассматриваться как достоверный и достаточный для представления информации пользователю, так как информация, полученная путем распознавания отдельного кадра, очень сильно зависит от качества этого кадра и может быть искажена помехами: достаточно привести букву «О», которая при засвечивании может быть распознана как буква «С». Кроме того, имея только статический кадр, невозможно сделать каких-либо выводов о траектории и параметрах движения транспортного средства.

Статический уровень конфигурируется информационным графом, который обладает следующими свойствами: каждый элемент графа реализуется так называемым *фильтром*, который выполняет самостоятельную процедуру преобразования данных. На вход каждого фильтра подключается выход одного входящего фильтра, который должен выполняться раньше, или первичный источник данных; на выход фильтра может подключаться множество исходящих фильтров, которые выполняются следом. Данные принимаются от входящего фильтра в едином контейнере, который содержит индекс типа данных, ссылку на данные фильтра и ссылку на контейнер-предок, принятый от входящего фильтра.

Поскольку каждый фильтр выполняется в собственном потоке, этап преобразования данных, реализуемый фильтром, может быть выполнен с применением параллельных гибридных вычислений с использованием свободных библиотек CUDA или OpenCL, без необходимости разработки сложных алгоритмов синхронизации с другими потоками.

Граф разбора данных обладает следующими свойствами: выполняется в собственном потоке, имеет начало (вход) и конец выполнения (выход), которые определяют такт выполнения; в рамках одного такта фильтры выполняются последовательно (передавая данные со своего выхода на вход ожидающим фильтрам), асинхронно (создавая независимые друг от друга в рамках такта пути выполнения), но с учетом (при наличии) списка зависимостей от других фильтров (рис. 2).

Граф разбора данных может обрабатываться с использованием вычислительного конвейера, при этом начало выполнения очередного такта наступает до окончания выполнения предыдущего, что позволяет сократить среднее время обработки кадров.

Рис. 2. Структура графа разбора данных.

Степень загрузки конвейера ограничивается только необходимостью синхронизации процесса передачи результата на другой уровень. Однако большинство трудоемких операций (более 80% всех фильтров), начиная с получения очередного кадра до этапов нахождения контуров, можно выполнять, используя конвейерную обработку, не дожидаясь завершения процесса разбора очередного кадра, что ранее не использовалось в подобных программных системах (рис. 3).

Рис. 3. Схема вычислительного конвейера с использованием графа разбора данных.

Граф разбора данных может быть встроенным в приложение, представлен в виде подгружаемого по технологии plug-in модуля динамической библиотеки («.dll», «.so», «.dylib») или описан в формате XML-файла. Используемые графом фильтры могут быть встроенными в приложение или подгружаться в виде отдельных модулей.

Однозначная идентификация ГРЗ в соответствии с типом осуществляется с использованием информации о структуре и начертании номерных пластин, которая также описывается во встроенных или подгружаемых шаблонах номерных пластин.

Динамический уровень отвечает за выполнение следующих процедур:

добавление данных распознанных частично или полностью номеров в список номеров или рамок для накопления;

реализацию алгоритма слежения: накопление информации о перемещении рамок для предсказания направления движения и сопоставления с результатами распознавания на следующем временном шаге;

обновление значений достоверности распознавания и горизонтальной относительной ошибки позиционирования для каждого символа, а также значений достоверности распознавания пластины в целом;

настройку шага распознавания при отсутствии или наличии «движения» в кадре для ускорения процесса сканирования видеофайлов;

кэширование кадров с движением для сохранения по требованию на уровне представления, удаление сохраненных и старых кадров.

Динамический уровень исполняется в самостоятельном потоке, отличном от главного потока программы. Взаимодействие динамического уровня со статическим реализовано по принципу событийной модели, т.е. статический уровень реализует события, а динамический осуществляет подписку на их обработку. События поступают на динамический уровень синхронно, в хронологическом порядке во времени.

Включение в алгоритм *уровня представления* позволяет выводить (передать) пользователю не всю информацию, хранящуюся на динамическом уровне, а только удовлетворяющую заданным условиям, – например, критериям достоверности распознавания, в нужные промежутки времени. Использование данного уровня дает возможность оптимизировать процессы графического отображения информации пользователю, сохранять информацию в БД и передавать ее клиентским приложениям по подписке.

Также на данном уровне вычисляются параметры траектории движения ТС: средняя скорость на участке дороги, направление движения по отношению к плоскости наблюдения камеры, географическая координата и др.

Траектория движения вычисляется по видеопотоку с использованием предварительно заданных в программе, параметров камеры (метрических и пиксельных размеров матрицы камеры географической координаты ее размещения) и фактического фокусного расстояния объектива.

Пример функционирования программного комплекса

Программный комплекс разработан в среде быстрой разработки приложений Embarcadero RAD Studio. Графические преобразования над матрицами изображений и поиск контуров выполняются с использованием библиотеки компьютерного зрения OpenCV. Распознавание символов реализовано на основе шаблонного метода с применением собственных классов и оригинального растрового классификатора символов.

Программный комплекс тестировался с применением графа разбора данных, состоящего из 133 фильтров, каждый из которых выполнялся в собственном потоке, а общее количество потоков, одновременно используемых приложением, превышало 145.

Тестирование программного комплекса выполнялось в режиме реального круглосуточного использования для регистрации ТС, проезжающих через ворота крупного потребительского гаражного кооператива. Программа по видеопотоку с камеры, установленной на въезде в кооператив, распознавала номера, определяла направление, траекторию, среднюю скорость движения транспортного средства и сохраняла информацию в локальной БД. Для тестирования использовались две следующие аппаратные платформы:

1) рабочая станция на базе ЦП Intel Celeron G1840 2.80 ГГц (2 ядра, 2 потока) с оперативной памятью 4 ГБ; IP-камера Beward N35110 с разрешением видеопотока 1280x1024 и кодированием MJPEG;

2) рабочая станция на базе ЦП Intel Core i3 8100 3.6 ГГц (4 ядра, 4 потока) с оперативной памятью 8 ГБ; IP-камера Beward SV3210R с разрешением видеопотока 2560x1920 и кодированием H265. Тестирование производилось с заданием области интереса (region of interest, ROI), достаточного для распознавания номеров въезжающих ТС за воротами и выезжающих – перед воротами, а также определения средней скорости и направления движения на данном участке траектории.

В ходе тестирования были получены следующие результаты скорости сканирования: в первом случае – более 10 кадров в секунду, во втором – более 8,9 кадров в секунду (рис. 4). В том и в другом вариантах происходило гарантированное распознавание номеров с читаемыми символами, даже в случае временного перекрытия номерной пластины препятствием (человеком или другим ТС) или частичного перекрытия номерной пластины фаркопом ТС.

Из проведенных экспериментов можно сделать вывод, что реализованная технология обработки данных в многопоточном режиме позволяет успешно использовать программный комплекс на офисных компьютерах бюджетного уровня с многоядерными процессорами.

Рис. 4. Просмотр архива распознанных номеров в режиме сканирования с отображением вектора направления движения на участке траектории.

Заключение

Основным преимуществом предложенного подхода является равномерность загрузки всех вычислительных ресурсов компьютерной системы со стороны программного комплекса. Это позволяет даже на компьютере среднего уровня параллельно использовать необходимые для офисной работы пользовательские программы, – например, приложения Microsoft Office, при этом средняя загрузка всех ядер ЦП приложением в процессе распознавания номеров ТС может достигать 80%.

Ключевыми особенностями описанного алгоритма, отличающими его от используемых подходов, являются: применение информационного графа в сочетании с конвейерной обработкой для эффективного разделения потоков выполнения и распределения вычислительных ресурсов в процессе преобразования данных; разделение процесса обработки данных на три уровня, позволяющих выполнять накопление информации для реализации алгоритмов слежения, повышения достоверности информации, вычисления параметров траектории движения ТС, а также возможность предоставления информации клиентам по заданным критериям.

ЛИТЕРАТУРА

1. *Елизаров А.И., Афонасенко А.В.* Методика построения систем распознавания автомобильного номера // Известия Томского политехнического университета. – 2006 – № 8. – С. 118-122
2. *Обухов А.В., Ляшева С.А., Шлеймович М.П.* Методы автоматического распознавания автомобильных номеров // Вестник Чувашиского университета. – 2016 – № 3. – С. 201-207.
3. *Артёмьева И.Л., Тютюнник М.Б.* Схемы распараллеливания вычислений для системы конфлюэнтных продуктов // Информатика и системы управления. – 2005. – № 2(8). – С. 102-112.
4. Ramamoorthy C.V. Pipeline Architecture // ACM Computing Surveys (CSUR). – 1977. – Vol. 9, № 1. – P. 61-102.
5. *Krasner G.E., Pope S.T.* A cookbook for using the model-view controller user interface paradigm in Smalltalk-80 // The Journal of Object Technology. – 1988. – Vol. 1, № 2. – P. 26-49.
6. *Амосов О.С., Баена С.Г., Иванов Ю.С., Со Х.* Система автоматического управления шлагбаумом с использованием технологий нечеткого вывода и компьютерного зрения // Интернет-журнал «Наукovedение». 24.02.2017. URL: <http://naukovedenie.ru/PDF/42TVN117.pdf> (дата обращения: 28.07.19).

E-mail:

Петропавловский Михаил Вячеславович – mv2006-68@mail.ru;

Дубровин Владислав Львович – vldubrovin@yandex.ru.