О едином подходе к программной реализации фильтрации изображений по локальной окрестности

Юрин Д. В.
yurin_d@inbox.ru
Москва, ВМиК МГУ им. М. В. Ломоносова

Любая практически полезная система обработки изображений включает в себя, наряду с другими компонентами, разнообразные алгоритмы фильтрации изображений. Это могут быть алгоритмы подавления шумов, улучшения изображений, выделения характеристических особенностей, таких как края, уголки, дифференциальные инварианты, текстурные признаки. Часто фильтры используются последовательно, образуя «сложный» фильтр (СФ). Более того, многие широко распространенные фильтры, такие как детекторы краев Канни, DiZenzo или детектор уголков Харриса, по сути, тоже являются СФ, так как их эффективная реализация основана на свойстве сепарабельности функции Гаусса. Такие СФ можно представить в виде направленного ациклического графа, узлами которого являются элементарные фильтры (ЭФ), а ребрами (стрелками) представляются изображения, получаемые в результате обработки ЭФ, и передаваемые на вход следующего фильтра. На Рис. 1 представлен такой граф для детектора Харриса. Прямолинейная программная реализация таких фильтров обычно состоит в выделении памяти под каждое промежуточное изображение (ребро графа на Рис. 1), что приводит к чрезмерным затратам памяти, учитывая, что типичный размер изображения для любительских фотоаппаратов достигает 10 Мпикс., а в аэрокосмической отрасли перевалил за гигабайт. В то же время, многие из используемых фильтров работают по локальной окрестности точки, и для обработки отдельной строки изображения требуется знание только некоторой полосы вокруг этой строки на исходном изображении.

Постановка задачи

Целью работы являлась разработка программных библиотек для реализации разнообразных фильтраций изображений по локальной окрестности, обеспечивающих невысокие требования к объему оперативной памяти, возможно, ценой незначительного снижения быстродействия, простоту реализации новых фильтров и легкость адаптации к различным способам представления изображения в памяти, включая построчное чтение с жесткого диска.

Структура системы

Требования быстродействия предопределили выбор шаблонного проектирования на C++. Полиморфные методы могут вызываться только

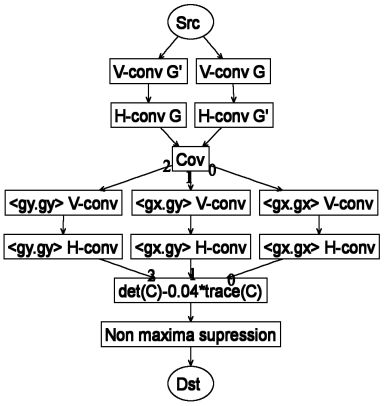

Рис. 1. Детектор характеристических точек Харриса.

на начальной и конечной стадии обработки, и не более нескольких раз на строку изображения. Каждый ЭФ реализуется в виде класса, производного от `FilterElementary`, фильтр имеет N входов (`class InPort`) и M выходов (`class OutPort`). Буфер с полосой изображения содержится в классе `OutPort`; для подключенного к `OutPort` класса `InPort` буфер доступен только для чтения, что обеспечивает существенную экономию в том случае, когда к выходу фильтра подсоединено много входов других фильтров. Задача дополнения изображения за его края (на размер требуемой окрестности) возложена на класс буфера. Для составления СФ создаются объекты — ЭФ, и выходы одних из них соединяются со входами других, типичный код выглядит как

```
filter1.out3.ConnectTo(&filter2.in);
```

Часто используемые СФ могут быть представлены в виде объекта `FilterComplex`, имеющего общую базу с `FilterElementary`. Источником данных является фильтр с $N = 0$, а приемником — с $M = 0$. Для приведения разработанного фильтра в действие создается объект `FiltersSystem`, через функцию `Assign` ему задаются источники и приёмники, после чего вызывается функция `Run()`. Система сама решает, когда передать управление (`processLine()`) каждому ЭФ на основе готовности входных данных и готовности подключенных фильтров принять данные. Реализация ЭФ состоит в переопределении функции `processLine()`, которой показываются необходимые фрагменты входных изображений в виде массивов, результат обработки полосы следует записать в выходной(ые) порт(ы) через предоставленный(ые) итератор(ы).

Важным компонентом системы является автоматическая настройка параметров и средства отладки. По вызову функции `Assign()` путем прослеживания связей от источника к приемнику с помощью поиска в глубину по графу находятся все задействованные ЭФ, отслеживаются возможные петли, если указано — результат немедленно сохраняется в формате `.dot`, по которому с помощью пакета `GraphViz` может быть визуализирован фактический граф системы как на Рис. 1. Вместе с диагностическими сообщениями это позволяет легко локализовать ошибку соединения фильтров. Повторным поиском от приемников к источникам достигается проверка, что нет неподключенных входов. Мультиграф фильтра сводится к графу и топологически сортируется (ТС). В порядке ТС для каждого фильтра инициализуются значения размеров полных входных изображений и предоставляется возможность вычислить необходимый размер локальных окрестностей для обработки. Теперь можно определить высоту необходимых буферов. Однако, проблемой является то, что размер запрашиваемой локальной окрестности по высоте одновременно является величиной задержки, с которой фильтр будет обрабатывать строку с данным номером. Чтобы система функционировала, должно выполняться условие равенства задержек по всем ветвям входящим в данный узел (фильтр). Такая задача широко известна в области конструирования микросхем [1] как задача балансировки графа информационных потоков, и, после некоторой адаптации, эти методы могут быть с успехом здесь применены, как для балансировки, так и для минимизации суммарного объема буферов в системе в целом. Применяемые алгоритмы также основаны на ТС графа.

Заключение

Тестирование разработанной системы показало, что накладные расходы по времени чрезвычайно малы, структура программы становится прозрачной, а затраты памяти — приемлемыми. Испытания, проведенные на студентах показали, что использование разработанной библиотеки не представляет проблем и ведет к повышению качества кода и ускорению разработки как ЭФ, так и СФ.

Работа выполнена при поддержке РФФИ, проект № 06-01-00789-а.

Литература

[1] *Chatterjee M., Banerjee S., Pradhan D. K.* Buffer Assignment Algorithms on Data Driven ASICs // IEEE Transactions on computers. — January 2000. — V. 49, No. 1. — P. 16–32.