УДК 519.714.2
ББК 32.965.5

ВЕКТОРИЗАЦИЯ РАСТРОВЫХ ИЗОБРАЖЕНИЙ

Гусев С.С.[1]
(*ФГБУН Институт проблем управления им. В.А. Трапезникова РАН, Москва*)

Данная работа посвящена разработке средства для векторизации цветных растровых изображений. Цель работы – разработка приложения для перевода цветной растровой графики в векторный формат. В работе описывается алгоритм векторизации растровых изображений Potrace, приводятся описания векторных форматов Scalable Vector Graphics (SVG), Vector Markup Language (VML), Postscript, Adobe Portable Document Format (PDF), Adobe Illustrator.

Ключевые слова: векторная графика, растровая графика, векторизация, формат, алгоритм, программа.

1. Введение

Черно-белые изображения могут быть представлены как точечный или векторный рисунок.

Точечный рисунок представляет изображение как сетку черных или белых пикселей. Векторное изображение можно описать через алгебраическое выражение его контуров в форме кривых Безье (Bezier). Преимущество векторного изображения – это возможность масштабирования до любого размера без потери качества. Векторные изображения независимы от разрешения любого специфического устройства. Они особенно популярны в очертании шрифтов, которым можно придать различный размер, например форматы векторных шрифтов, включающих шрифт PostScript Type 1, TrueType, и Metafont. Кроме того, это

[1] Сергей Сергеевич Гусев, соискатель (gs-serg@mail.ru).

очень актуально в загружаемых и выгружаемых устройствах типа сканеров, мониторов, принтеров, в конечном счете устройств, связанных графикой.

Процесс преобразования векторного изображения в точечное называют *визуализацией (rendering)*. Обратный процесс превращения точечного изображения в векторное - *трассировка (tracing)* или *векторизация*.

Ясно, что никакой алгоритм трассировки не может быть совершенен в абсолютном смысле, так может быть много всевозможных контуров, которые в конечном итоге могут привести к такому же самому точечному рисунку. Самые различные контуры могут быть применены к исходному точечному рисунку, они могут быть сходными с оригиналом или выглядеть более эстетично, нежели другие. Обычный способ визуализации точечных рисунков с высоким разрешением состоит в том, чтобы прорисовать каждый черный пиксель как точный квадрат, но это иногда приводит к контурным неровностям или так называемым «лестничным ступенькам». Ясно, что неровности [1] плохо смотрятся, не смотря на приближенность изображения к оригиналу. Нет никакой абсолютной уверенности в том, что алгоритм трассировки будет оптимальным, но ясно, что именно такие алгоритмы дают лучшие результаты, чем подобные.

Среди алгоритмов векторизации выделяется алгоритм Potrace, который обладает большей точностью и быстродействием по сравнению с другими алгоритмами. Но недостатком данного алгоритма является отсутствие возможности векторизовать цветные растровые изображения. К моменту начала работы существовала первая версия приложения PotraceGUI, расширяющего возможности алгоритма Potrace, которая позволяла векторизовать цветные растровые изображения, сохраняя полученный результат в SVG-формате. В ходе работы над программой она была улучшена, была расширена ее функциональность.

2. Постановка задачи

В 2002 году была создана программа, позволяющая векторизовывать цветные растровые изображении. Недостатками этой программы являлись: взаимодействие с векторизующим модулем через командную строку, поддержка только одного выходного формата, невозможность управления настройками алгоритма векторизации.

Новая версия программы должна давать возможность пользователю:

- открывать, просматривать и получать информацию об открытом растровом файле;
- уменьшать цветность, переводить в режим представления оттенками серого цвета, переводить в монохромный режим, отрытый растровый файл;
- сохранять открытый растровый файла с именем, отличным от исходного;
- настраивать параметры векторизации;
- векторизовывать открытый растровый файл
- сохранять результат векторизации в выбранном пользователем векторном формате;
- открывать, просматривать и сохранять в выбранном формате, созданный ранее векторный файл.

Для реализации этого функционала приложение должно:

- иметь средство отображения и сбора информации растрового файла;
- иметь средство изменения цветового режима растрового файла;
- иметь средство отображения SVG-файла;
- иметь средство конвертирования, записи и чтения векторных форматов.

2.1. ОСНОВНЫЕ РЕШЕНИЯ ПО СРЕДЕ РАЗРАБОТКИИ СОСТАВУ ПРИЛОЖЕНИЯ

В качестве среды разработки приложения решено использовать Borland Delphi 7.

Для работы с растровыми файлами удобно использовать библиотеку Graphics32, которая позволяет удобно их просматривать и редактировать.

Для просмотра SVG-файлов на настоящий момент практически единственной альтернативой является Adobe SVGViewer 3.0. Данный ActiveX модуль можно включить в среду Delphi, что позволяет просматривать векторные файлы формата SVG-файлы из разрабатываемого приложения.

Для поддержки дополнительных форматов решено использовать библиотеку PsToEdit, которая поддерживает конвертирование векторных файлов формата Postscript в ряд других форматов.

Для редактирования и компилирования библиотеки PsToEdit необходимо использовать среду разработки Microsoft Visual Studio .NET 2003.

В результате имеется следующий перечень ПО, используемого при разработке приложения:

- Delphi 7;
- Microsoft Visual Studio .NET 2003
- Adobe SVGViewer 3.0;
- PsToEdit;

2.2.ТРЕБОВАНИЯ К ИНТЕРФЕЙСУ ПОЛЬЗОВАТЕЛЯ

Требования к интерфейсу приложения на этапе проектирования можно свести к следующему набору требований:

- приложение должно иметь пользовательский интерфейс характерный для большинства приложений Windows, т.е. требуется наличие главного меню, панели инструментов;
- приложение должно обеспечивать визуальное отображение исходного (растрового) и результирующего (векторного) изображений.

2. Алгоритм векторизации в приложении Potrace

Алгоритм potrace разработан для работы [2] с изображениями высокого разрешения. Сложные изображения, отскани-

рованные с высоким разрешением можно преобразовать векторный контур, к примеру, логотип компании или университетская эмблема. Другое возможное применение – это преобразование растровых шрифтов в контурные, если первоначальные растровые шрифты будут с высокой, достаточной разрешающей способностью. Никакой алгоритм трассировки не будет работать хорошо, если используется очень маленький масштаб изображения, рисунок 1, a, b, c, d.

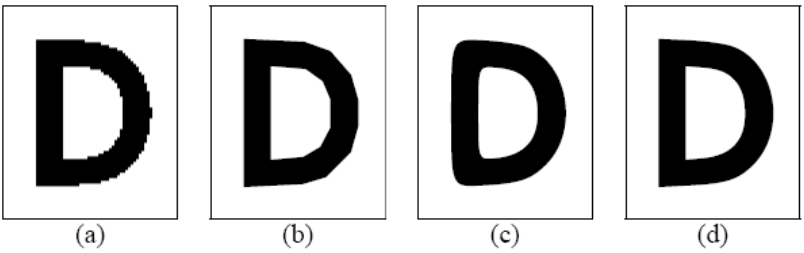

Рис. 1. Обнаружение угла. (a) оригинальный точечный рисунок; (b) очень много углов; (c) очень мало углов; (d) оптимальное обнаружение угла

Это точечные рисунки для типичного 10pt экранного шрифта, представленного в разрешении 75 dpi (точек на дюйм). Однако, этого достаточно для трассировки неточных форм, такого как отсканированного, написанного от руки текста или рисунков мультипликации, если даже если они имеют относительно умеренные разрешающие способности.

Каждый оптимальный этап алгоритма трассировки должен выполнять несколько функций. Две из этих функций выполняют поиск самой вероятной кривой, которая приближает данные контуры к искомым обнаруженным углам. Существует связь между этими двумя функциями. Если обнаружено слишком много углов, результат трассировки будет напоминать не гладкий многоугольник. Если также при обнаружении углов их будет немного, то контур рисунка будет выглядеть гладким, но округленным. Пример показанный на рисунке 1.

Другая важная функция, выполняемая алгоритмом трассировки должна определить, какая особенность растрового изображения уместна, и какие есть особенности артефактов (искажений) сканирования или процесса визуализации. Те особенности, которые можно считать артефактами должны быть полностью отфильтрованы, потому что, если даже небольшое искажение остается, это может привести к видимым недостаткам по окончании трассировки. При этом прямая линия будет рассматриваться как выпяченная, очень маленькая и наклонная. Когда изображение представлено как точечный рисунок, такая линия будет в виде лестницы, где отдельные «шаги ступеньки» могут быть далеко друг от друга. Независимо от того, как далеко эти шаги отдалены друг от друга, на выходе должна быть прямая линия, иначе это вызовет визуальное искажение.

Алгоритм potrace преобразовывает точечный рисунок в векторный контур за несколько этапов. Первоначальный этап – это расчленение точечного рисунка на несколько путей, которые формируют границы между черно-белыми областями.

На втором этапе, каждый путь приближен к оптимальному многоугольнику. На третьем этапе, каждый многоугольник преобразован в сглаженную кривую. На четвертом (необязательном) этапе, замыкающая кривая оптимизирована и соединяет вместе последовательные кривые Безье там, где это возможно. Наконец, результат сгенерирован в требуемый формат. Следующие подразделы описывают каждый из этих этапов более подробно.

3.1. ПУТИ. РАЗБИВКА ПУТИ

Potrace использует следующий прямой метод анализа точечного рисунка в контуры. Сначала, выбирается пару смежных пикселей различного цвета [3]. Это достигается, к примеру, выбором крайнего левого черного пикселя в каком-нибудь ряду. Соединение двух выбранных пикселей формирующих край осуществляется таким образом, чтобы черный пиксель был слева, а белый пиксель справа. Этот край определяет длину пути.

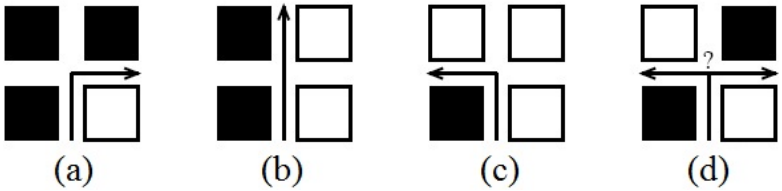

Рис. 2. Алгоритм разбивки пути

Дальнейшее расширение этого пути происходит таким образом, что каждый новый край имеет черный пиксель слева и белый пиксель - справа, относительно направления пути. Другими словами, проходя по граням между пикселями и каждый раз доходя до угла, выбирается либо прямое направление, либо осуществляется поворот между левым или правым пикселем, в зависимости от цвета окружающих пикселей, как показано на рисунке 2.2. Это происходит до тех пор, пока путь не дойдет до вершины, от которой он начинался, к точке, которая определяет закрытый путь.

Каждый раз, когда определен закрытый путь, из графического контурного изображения удаляются и инвертируются все цвета пикселя в его внутренней области. Это определяет новый точечный рисунок, к которому применяется рекурсивный алгоритм к оставшимся там некоторым черным пикселям. В итоге образуются закрытые пути, которые будут переданы к следующей стадии алгоритма potrace. В более поздних стадиях алгоритм potrace рассматривает каждый путь самостоятельно.

3.2. РАСЧЕТ ПОВОРОТА

В ситуации, описанной на рисунке 2, d), указывается расчет того, произойдет ли левый поворот или правый поворот. Этот расчет не влияет на положительный или отрицательный результат алгоритма декомпозиции пути, поскольку любой путь будет замкнутым. Однако, такой расчет оказывает влияние на форму закрытых выбранных путей.

В алгоритме potrace, варианты левого или правого поворота осуществляются *расчетом поворота*, который определяется через опцию командной строки называемой turnpolicy .

Возможные расчеты поворота:

- *левый*, который всегда производит левый поворот;
- *правый*, который всегда производит правый поворот;
- *черный*, который предпочитает подключать черные компоненты пиксельного изображения;
- *белый* -белые компоненты;
- *меньшинство*, который предпочитает подключить черный или белый цвет, являющийся наименее выраженным и находящийся в пределах данного соседства с текущей позицией;
- *большинство*, который предпочитает подключить цвет, являющийся наиболее выраженным,
- *случайный*, который производит случайный выбор цвета.

Заданный по умолчанию вариант поворота - *меньшинство*.

3.3. ОПТИМИЗАЦИЯ ПУТИ

Удаление пятен. Удаление пятен выполняется, разбросом всех путей, внутренняя область которых имеет меньше чем t пикселей, для данного параметра s. Параметр t может быть представлен через опцию командной строки называемую turdsize

Многоугольники. Итогом работы данного этапа есть оптимальный многоугольник, имеющий сходство с очертаниями пути. В значении «оптимальный» мы подразумеваем точный, а не приближенный.

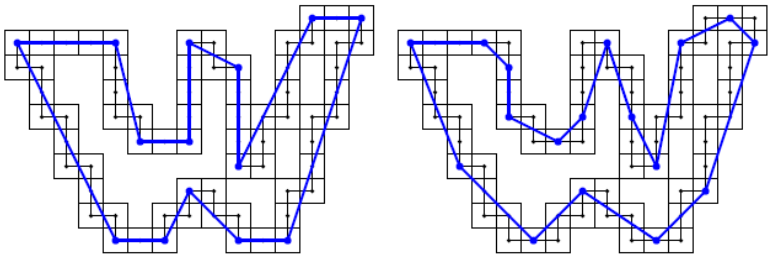

Рис. 3. Оптимизированный и неоптимизированный многоугольник пути

3. Получение векторного контура

От многоугольников до векторных контуров. Конечная стадия алгоритма - преобразование многоугольника, полученного в предыдущей стадии в сглаженный векторный контур. На предварительном этапе, происходила коррекция позиции вершины многоугольника для того, чтобы было наилучшее соответствие первоначальному точечному рисунку насколько это возможно. На главном этапе, определяются углы и кривые, основанные на длинах смежных сегментов линии и углах между ними.

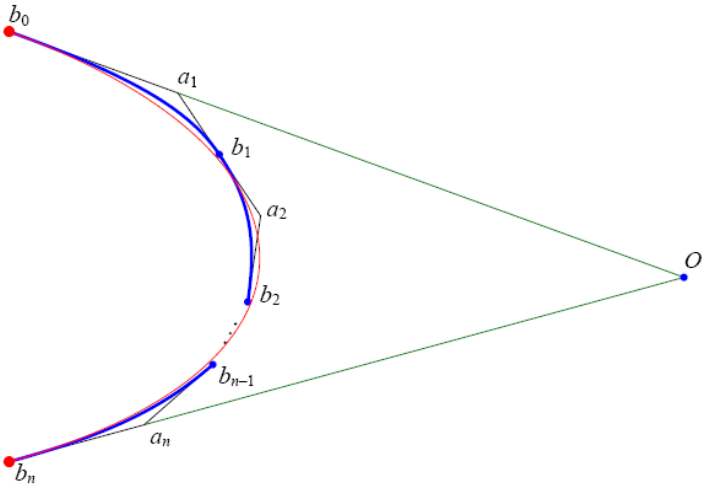

Рис. 4. Оптимизация кривой

Оптимизация кривой. Результатом предыдущей стадии алгоритма potrace, после анализа угла и сглаживания, является кривая, состоящая из сегментов кривой Безье и сегментов прямой линии. Результирующая кривая очень близка к конечному результату алгоритма potrace. Однако, есть еще одна

дополнительная, последняя стадия алгоритма, стадия оптимизации кривой, которая делает попытку дальнейшего её улучшения, соединяя смежные сегменты кривой Безье вместе там, где это возможно. Оптимизация кривой [4] делает незначительные изменения форм конечной кривой и они настолько маленькие, что различий обычно не видно. Однако, окончательная кривая состоит из меньшего количества сегментов и таким образом может быть представлена более сжато в финальной стадии программы. Если оптимизация кривой не желательна, это можно заблокировать запуском опции командной строки алгоритма potrace, называемой long-curve.

4. Генерация результата

Масштабирование и вращение. Алгоритм potrace производит семейство кривых, каждая из которых состоит из сегментов кривой Безье и сегментов прямой линии. Конечные и контрольные точки их сегменты являются произвольными точками в координатной плоскости [5]. В зависимости от выбранного ввода и параметров, алгоритм potrace выполнит линейное преобразование (масштабирование изображения к желательному размеру, и возможно его вращать).

Кодирование избыточной информации. При использовании одного из PostScript алгоритма (PostScript или EPS), potrace использует очень уплотненный числовой формат, чтобы преобразовать результат кривых Безье. Для этого требуется много параметров кривой. В принципе необходимо 6 параметров для описания каждого сегмента кривой Безье (одна конечная точка и две контрольных точки). Однако, если параметров мало, potrace может кодировать каждый сегмент, используя только 3 - 4 реальных числа.

Кодирование избыточности параметров кривых Безье всегда выполняется со встроенным алгоритмом PostScript, который использует макро возможности языка PostScript. Также кодиро-

вание избыточности параметров может быть выключено опцией командной строки longcoding и происходит дольше по времени, но имеет более приемлемый результат.

Квантование. Для большинства вычислений, конечные координаты, которые являются реальными числами, квантуются и округляются 1/10 пикселя. Таким образом, номер десятичных цифр необходимо представлять как каждую уменьшенную координату для эффективного размещения всех контрольных точек на сетке координат. Координаты точек можно тогда представлять как целые числа.

Заданная по умолчанию константа квантования 1/10 обычно дает хорошие результаты, однако это происходит с перестраиванием конфигурации через опцию командной строки - unit.

5. Пример окончательного результата алгоритма программы

Пример окончательного выполнения окончательного алгоритма potrace показан на рисунке 5.

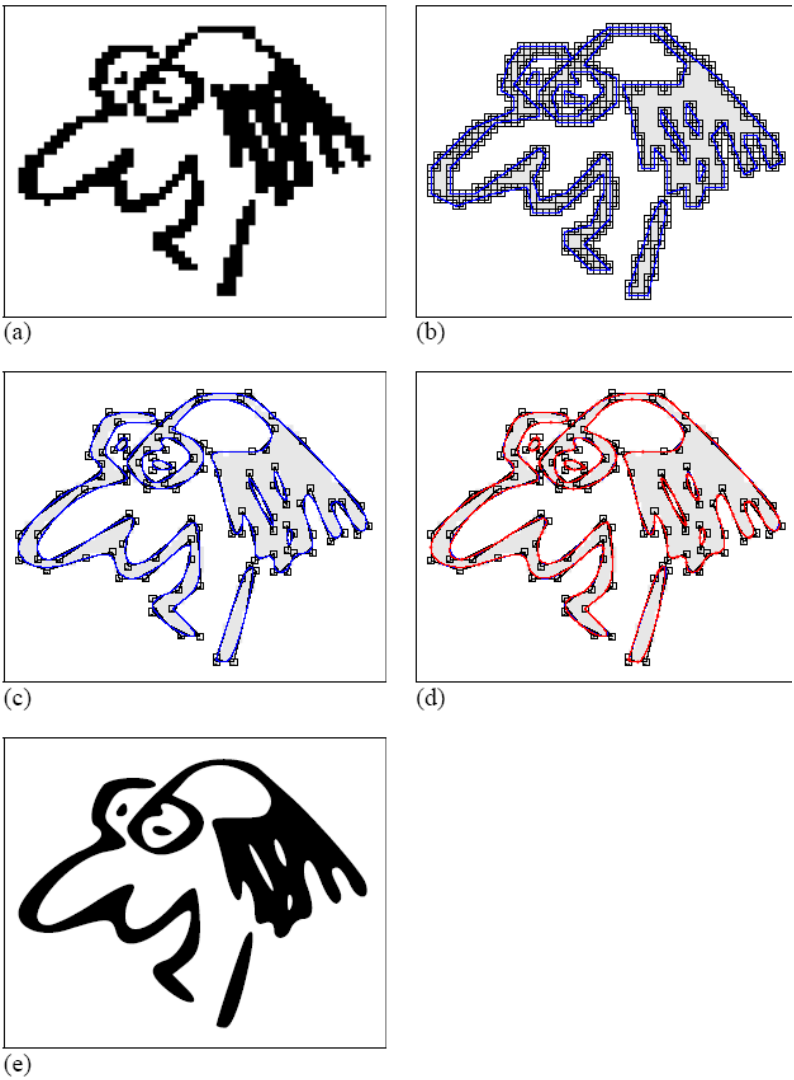

Рис. 5. Пример окончательного результата алгоритма программы. (a) первоначальный точечный рисунок, (b) декомпозиция пути и оптимальный многоугольник, (c) настройка вершины, анализ угла, и сглаживание, (d) оптимизация кривых, (e) конечный результат

Рисунок 5, a) показывает первоначальное точечное изображение. На рисунке 5, b) обратим внимание, как заданное по умолчанию «меньшинство» просчитывает поворот, сохраняя черные контуры по внешней стороне замкнутой фигуры, в тоже время сохраняет белые области в месте прорисовки волос на рисунке. Также отметим то, что области волос на рисунке было удалено пятнышко размером с единицу. Рисунок 5, b) также показывает оптимальный многоугольник, рассчитанный для каждого компонента пути. Рисунок 5, c) показывает откорректированные вершины многоугольника, относительно оригинального точечного рисунка, который показан серым цветом. Каждая вершина окружена ее квадратным модулем. На этом этапе происходит анализ углов для данного точечного рисунка, но обратим внимание, что только очень немногие углы обнаружены. Вообще анализ угла происходит лучше при более высоком разрешении точечного рисунка.

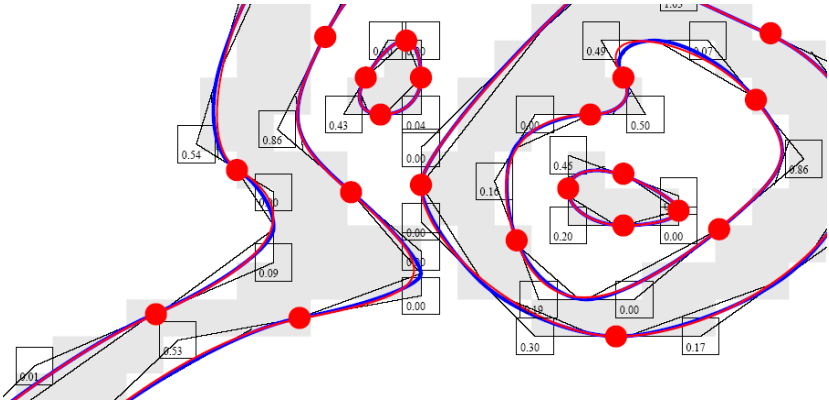

Рис. 6. Увеличенный, комбинированный фрагмент рисунка 5

На рисунке синим цветом показаны сглаженные кривые, замыкающие сегменты кривой Безье и сегменты линий. Рисунок 5, d) показывает результат оптимизации кривой: первоначальная

кривая - показана синим цветом, и оптимизированная кривая показана красным цветом. Красные точки являются индикаторами новых границ сегмента, при этом количество сегментов уменьшилось от 112 до 68, то есть на 40%. Конечный результат алгоритма показан на рисунке 5, e).

Отладка результата на рисунке 5, b), c), d) может быть произведена за счет различных вариантов командной строки, используемых в potrace - от d1 до d3.

6. Векторные форматы

6.1. ФОРМАТ SVG. ОБЩИЙ ОБЗОР

Формат SVG (Scalable Vector Graphics - масштабируемая векторная графика) - это новый XML-словарь, предназначенный для описания двухмерной векторной графики для Интернета и других приложений.

SVG был разработан Консорциумом W3C. Консорциум W3C - некоммерческая организация, занимающаяся разработкой открытых стандартов, например, HTML и XML, а также многих других важных словарей. В создании SVG участвовало более двадцати организаций, включая Sun Microsystems, Adobe, Apple, IBM и Kodak.

Компания Sun с самого начала принимала участие в разработке спецификации SVG и в рабочей группе экспертов имеется два активных ее представителя.

6.1.1. ПРЕИМУЩЕСТВА SVG

SVG имеет много преимуществ перед другими форматами изображений, например, перед форматами JPEG и GIF, которые являются наиболее распространенными форматами в Интернете.

Это чисто текстовый формат - файлы SVG могут быть прочитаны и модифицированы широким кругом приложений и обычно меньше по размеру, чем сравнимые изображения в форматах JPEG или GIF.

Масштабируемость - в отличие от растровых форматов GIF или JPEG, SVG является векторным форматом. Это означает,

что изображения SVG могут выводиться с высоким качеством при любом разрешении без возникновения эффекта «ступенек», который можно наблюдать при распечатке растровых изображений.

Возможность увеличения - вы можете увеличить любую часть изображения SVG без потери качества.

Текст в графике SVG находят поисковики и его можно выделять - в отличие от растровых изображений, текст в SVG можно выделять и можно находить с помощью поисковых машин. Можно искать заданную строку текста - например, имя города на SVG-карте.

Применение скриптов и анимация - SVG позволяет создавать динамичную и интерактивную графику, гораздо более изощренную, чем можно сделать с помощью растровых изображений или даже Flash-приложений.

Совместимость с технологией Java - SVG совместим с высокотехнологичным Java-графическим движком Java 2D API.

Это открытый стандарт - SVG является открытой рекомендацией. В отличие от других форматов, SVG не является чьей-либо собственностью.

Это чистый XML - будучи форматом XML, SVG предоставляет все преимущества XML:

- Возможность работы в различных средах.
- Интернационализация (поддержка Unicode).
- Широкая доступность для различных приложений.
- Легкая модификация через стандартные API, например, DOM API.
- Легкое преобразование таблицами стилей XSLT.

6.1.2. ОБЛАСТЬ ПРИМЕНЕНИЯ

Поскольку SVG является форматом XML, SVG-графика может легко генерироваться веб-серверами «на лету» с использованием стандартных средств XML, многие из которых написаны на Java. Например, веб-сервер может генерировать высококачественные и при этом небольшие по объему графики на основе оперативных биржевых сводок.

SVG позволяет создавать графику с помощью специальных графических пакетов (см. список приложений SVG) или автоматически (например, используя JavaServer Pages). SVG позволяет легко манипулировать графикой с помощью стандартных инструментов XML.

Возможность работы в различных средах - ключевое преимущество SVG. Например, с помощью генератора двухмерной графики SVG любое Java-приложение может экспортировать графику в формате SVG. И затем эти изображения могут импортироваться, просматриваться и модифицироваться в различных средах.

6.1.3. ТРАЕКТОРИЯ В SVG

Траектории в SVG представляют контуры объектов с помощью элемента «<path>». В атрибуте «d»сначала задается инструкция «M», которая дает указание сделать сдвиг к новой точке, от нее начнется кривая. Заглавная «M» в данном случае означает, что в описании используются абсолютные координаты, а маленькая «m» - что используются относительные (это относится и к другим инструкциям). После того, как с помощью инструкции «m» определена начальная точка траектории, используется либо инструкция «l», которая проводит линию до точки (координаты X Y этой точки задаются в качестве параметра для данной инструкции), либо инструкция «c», которая задает сегмент кривой Безье (в качестве параметра для этой инструкции задаются координаты опорных точек и координаты конечной точки). После выполнения одной инструкции q, текущая точка переместится в соответствии с координатным параметром. Атрибут «z», заданный в конце элемента «<path>» закрывает траекторию.

<path d=«M300 3225 l0 -3075 2325 0 2325 0 0 1313 0 1313 553 549 553 549 -32 275 c-43 370 -274 623 -611 669 -171 23 -251 12 -315 -46 m600 2741 c0 -50 -210 -266 -259 -266 -23 0 -41 67 -41 150 0 133 17 150 150 150 82 0 150 -15 150 -34

z»/>

Для группировки нескольких траекторий используется оператор «<g>». Атрибут «scale» данного оператора задает масштаб по вертикале и горизонтали, атрибут «fill» задает цвет, которым будут залиты контуры траекторий, которые относятся к данной группы. Цвет заливки задается в шестнадцатеричном формате (#AARRGGBB, где AA – величина альфа-канала, RR, GG, BB – величины красного, зеленого и синего каналов соответственно).

6.2. ФОРМАТ POSTSCRIPT

Postscript был разработан Джоном Уорноком и Чаком Гешке из Adobe Systems в начале 80-х гг. Исходно Postscript использовался как ядро механизма печати компьютеров Apple, но вскоре стал широко распространенным стандартом для большинства компьютерных систем. Интерпретаторы Postscript (в виде программных или аппаратных компонентов) для печати документов присутствуют практически во всех современных компьютерных системах. В Postscript используется модель изображения текста (или рисунков) на чистой странице. Когда страница готова, она выводится на печать и начинается «прорисовка» изображения очередной страницы. Это есть не что иное, как метод компиляции. Каждый документ Postscript включает в себя программу, которая печатает на принтере (или отображает на экране монитора) следующие друг за другом страницы.

6.2.1. ОБЛАСТЬ ПРИМЕНЕНИЯ

PostScript соединил в себе лучшие возможности принтеров и плоттеров. Подобно плоттерам, PostScript предоставляет возможность вывода высококачественной векторной графики и единый язык управления, который может быть использован любым производителем принтеров. Подобно матричным принтерам, PostScript предлагает удобные возможности по печати растровой графики и текста. В отличие от тех и других, PostScript может совмещать все эти типы вывода на одной

странице, давая намного больше гибкости, чем до этого имел любой принтер или плоттер.

PostScript — больше, чем типичный язык управления принтером, он является полнофункциональным языком программирования. Многие прикладные программы могут преобразовать документ в PostScript-программу, при выполнении которой будет получен начальный документ. Эта программа может быть послана непосредственно на принтер с поддержкой PostScript или преобразована интерпретатором PostScript в другой формат (для принтеров без поддержки PostScript), или результат её выполнения интепретатором может быть показан на экране. Так как исходная PostScript-программа одна и та же, PostScript называется *независимым от устройства*.

Большинство высокопроизводительных принтеров и плоттеров имеют встроенный интерпретатор языка PostScript. В то же время, простые принтеры домашнего класса поддерживают только элементарные графические операции, поэтому задача создания растрового изображения возлагается на центральный процессор. Существуют интерпретаторы языка PostScript для различных операционных систем, наиболее известный из них — свободная программа Ghostscript.

6.2.2. ЯЗЫК

Постскрипт — полнофункциональный язык программирования. Хотя программы на Постскрипте и создаются в основном не людьми, а другими программами, в принципе ничто не мешает писать на нём программы для обсчёта графики, реализации численных методов решения математических задач и т. п.

Постскрипт — интерпретируемый стековый язык, похожий на Форт. Синтаксис языка использует обратную польскую нотацию, что делает ненужным использование скобок, однако требует некоторой практики для чтение текста программы из-за необходимости держать в голове содержимое стека. Большинство операторов берут операнды со стека и помещают результат вычислений на стек. Литералы (строки и числа) помещают свою копию на стек.

PostScript имеет черты метафайла, совмещая поддержку как векторных, так и растровых изображений. Шрифты в PostScript только векторные. Положение элементов на странице задаётся в типографских пунктах.

6.2.3. ТРАЕКТОРИИ В POSTSCRIPT

Траектории в Postscript описывают контуры объектов. Цвет, которым будет заполнен контур, указывается оператором «R G B setrgbcolor», где R, G и B - интенсивность красного, зеленого и синего цветов соответственно (0 – минимальная интенсивность, 1 – максимальная). Начало описания очередного контура обозначается оператором «newpath», далее следует список операторов, по одному на строку. Вначале этого списка следует оператор «moveto», который определяет координаты начальной точки траектории (формат – «X Y moveto»). Затем - оператор «lineto», который проводит линию до определенной точки (формат – «X Y lineto») или оператор «curveto», рисующий кривую Безье (формат – «X1 Y1 X2 Y2 X3 Y3 curveto», первые две указанные точки - опорные, третья – конечная точка, начальной точкой является крайнее положение предыдущей фигуры). Замыкание контура и его заливка осуществляются командой «closepath fill».

6.3. ФОРМАТ VML

VML (Vector Markup Language) — это XML-приложение, разработанное корпорацией Microsoft для кодирования двумерной векторной графики. Поддержка VML в настоящее время осуществляется только обозревателем Internet Explorer 5.x и пакетом Microsoft Office 2000.

Microsoft передал в мае 1998 г. спецификацию этого языка W3C в качестве предложения для стандартизации, однако W3C предложил в качестве стандарта собственную спецификацию SVG (Scalable Vector Graphics). Полной программной реализации SVG пока нет, но этот язык несомненно будет в ближайшее время поддержан многими системами.

Приведенное ниже описание VML соответствует его описанию в MSDN Library. Мы описываем только те возможности

VML, которые поддерживаются Веб-обозревателем; расширения VML, добавленные в Microsoft Office 2000, не рассматриваются. Для того, чтобы видеть, как элементы VML отображаются на экране, эту и последующие главы следует просматривать в обозревателе Internet Explorer 5.x.

6.3.1. ВКЛЮЧЕНИЕ VML В HTML-ДОКУМЕНТЫ

Как уже было сказано VML — это XML-приложение, т. е. специализированный язык, построенный по правилам XML. Для включения элементов VML в свою HTML-страницу мы должны сделать две вещи.

Во-первых, нужно задать пространство имен VML, чтобы анализатор HTML мог понять, когда ему вызывать XML-процессор для разбора конструкций VML. Для этого тег HTML нашей страницы нужно записать так:

<html xmlns:v=«urn:schemas-microsoft-com:vml»>

Теперь префикс v: в теге будет означать, что этот тег описывает VML-, а не HTML-элемент.

Во-вторых, необходимо указать обозревателю, как отображать обнаруженные в документе элементы VML. VML реализован в Microsoft Internet Explorer как встроенная реакция (default behavior). Поэтому в заголовок документа нужно включить следующие строки:

```
<style>
v\:* { behavior: url(#default#VML) }
</style>
```

Эта таблица стилей указывает, что теги с префиксом v: обрабатываются встроенной реакцией VML.

6.3.2. ТРАЕКТОРИИ В VML

VML имеет всего два основных элемента: «shape» и «group». Элемент «shape» описывает отдельную графическую фигуру, а элемент «group» позволяет объединять несколько фигур в группу с тем, чтобы в дальнейшем применять к этой группе различные преобразования. Элемент «shape» имеет атрибут «path», который задает форму фигуры как набор отрез-

ков и плавных кривых, после того, как с помощью инструкции «m» данного атрибута определена начальная точка траектории, используется либо инструкция «l», которая проводит линию до точки (координаты X Y этой точки задаются в качестве параметра для данной инструкции), либо инструкция «c», которая задает сегмент кривой Безье (в качестве параметра для этой инструкции задаются координаты опорных точек и координаты конечной точки). Так же в элементе «shape» задается атрибут «fillcolor», который отвечает за цвет, которым будет заполнена описанная в атрибуте «path» фигура. Цвет заливки задается в шестнадцатеричном формате (#AARRGGBB, где AA – величина альфа-канала, RR, GG, BB – величины красного, зеленого и синего каналов соответственно).

6.4. ФОРМАТ PDF

PDF (аббревиатура от англ. *Portable Document Format*) — кроссплатформенный формат электронных документов, созданный фирмой Adobe Systems с использованием ряда возможностей языка PostScript. В первую очередь предназначен для представления в электронном виде полиграфической продукции, — значительное количество современного профессионального печатного оборудования может обрабатывать PDF непосредственно. Для просмотра можно использовать официальную бесплатную программу Acrobat Reader, а также программы сторонних разработчиков. Для создания PDF-документов требуется собственническая shareware-программа Adobe Systems — Adobe Acrobat, либо программы сторонних разработчиков.

Формат PDF позволяет внедрять необходимые шрифты (построчный текст), векторные и растровые изображения, формы и мультимедиа-вставки. Поддерживает RGB, CMYK, несколько типов сжатия растровой информации. Имеет собственные технические форматы для полиграфии: PDF/X-1, PDF/X-3. Включает механизм электронных подписей для защиты и проверки подлинности документов. Имеется возможность импорта из большинства современных форматов текстовых документов, векторных и растровых графических форматов. В этом формате

распространяется большое количество сопутствующей документации.

6.4.1. ТРАЕКТОРИИ В PDF

Траектории в PDF описывают контуры объектов. Цвет, которым будет заполнен контур, указывается оператором «R G B rg», где R, G и B - интенсивность красного, зеленого и синего цветов соответственно (0 – минимальная интенсивность, 1 – максимальная). Далее следует список операторов, по одному на строку. Вначале этого списка следует оператор «m», который определяет координаты начальной точки траектории (формат – «X Y m»). Затем - оператор «l», который проводит линию до определенной точки (формат – «X Y l») или оператор «c», рисующий кривую Безье (формат – «X1 Y1 X2 Y2 X3 Y3 c», первые две указанные точки - опорные, третья – конечная точка, начальной точкой является крайнее положение предыдущей фигуры). Замыкание контура осуществляются командой «h», заливка – командой «f».

6.5. ФОРМАТ ADOBE ILLUSTRATOR

Adobe Illustrator, первоначально разработанный для платформы Macintosh, – известная и широко используемая программа создания изображений. Существуют версии для Macintosh, Microsoft windows и NeXT. Мощные возможности Adobe Illustrator обусловлены тем, что в качестве графических объектов здесь реализованы кривые Безье, а также наличием простого пользовательского интерфейса, который обеспечивает точное позиционирование сплайновых графических объектов. Использование кривых Безье дает некоторые преимущества при моделировании естест-венных (а в определенных случаях и искусственных) объектов, файлы Adobe Illustrator применяются для обмена графическими элементами.

Формат AI инкапсулирует и формализует в структурированном файле подмножество языка описания страницы (PDL) PostScript. Такие файлы предназначены для отображения на принтере PostScript, но могут включать и растровую версию

изображения, обеспечивая тем самым его предварительный просмотр. PostScript в полной реализации представляет собой мощный и сложный язык и способен определять почти все, что может быть отображено на двумерном устройстве вывода, формат AI адаптирован для хранения традиционных графических данных: рисунков, чертежей и декоративных надписей. Отметим все же, что файлы AI могут быть очень сложными. Мощь PostScript обусловлена в основном возможностью определять последовательности операций и затем объединять их простыми синтаксическими средствами. Эта скрытая сложность в файлах Adobe Illustrator иногда (но не всегда) сводится к минимуму.

Простые файлы AI конструировать довольно легко, и прикладная программа сможет создавать файлы, которые будут прочитаны любой программой чтения AI и распечатаны на любом PostScript-принтере. А вот чтение файлов AI – совсем другое дело. Некоторые операции могут оказаться слишком сложными для реализации и моделирования программой визуализации. Поэтому разработчики часто предпочитают не визуализировать изображение из данных этого подмножества PostScript. Тем не менее следует отметить, что, как правило, почти все изображение можно реконструировать простыми операциями. Если хотите разработать программу чтения файлов Adobe Illustrator, то рекомендуем в качестве подсказки воспользоваться исходными текстами системы GNU GhostScript, которая содержит почти полную реализацию языка PostScript.

6.5.1. ТРАЕКТОРИИ В ФОРМАТЕ ADOBE ILLUSTRATOR

Траектории в данном формате описывают контуры объектов. Цвет, которым будет заполнен контур, указывается оператором «C M Y K k», где C, M, Y и K - интенсивность цветов цветовой модели CMYK. Начало описания очередного контура обозначается оператором «*u», далее следует список операторов, по одному на строку. Вначале этого списка следует оператор «m», который определяет координаты начальной точки траектории (формат – «X Y m»). Затем - оператор «l», который проводит линию до определенной точки (формат – «X Y l») или

оператор «c», рисующий кривую Безье (формат – «X1 Y1 X2 Y2 X3 Y3 curveto», первые две указанные точки - опорные, третья – конечная точка, начальной точкой является крайнее положение предыдущей фигуры). Замыкание контура и его заливка осуществляются командой «f».

7. Заключение

В работе рассмотрен принцип векторизации растровых изображений, основные векторные форматы.

Разработана новая версия программы, векторизующей цветные растровые изображения, в которой были исправлены недостатки первой версии. Векторизующий модуль был встроен в программу, что позволило увеличить скорость получения результата более чем на порядок.

Реализована возможность сохранять результат в одном из нескольких векторных форматов, открывать сохраненные ранее результаты, уменьшать количество цветов исходного изображения, добавлена поддержка 32-х битных изображений.

Литература

1. ИВАНОВА Н.Ю., МАЛИНИН А.А., ТАЯНОВСКАЯ Ю.Б. *Метод инвариантного анализа изображений, заданных в векторной форме* // Научно-технический вестник информационных технологий, механики и оптики. – 2006. – С. 188 – 192.
2. ГОРОШКИН А.Н. *Применение векторного подхода к распознаванию рукописных символов* // Вестник Сибирского государственного аэрокосмического университета имени академика М.Ф. Решетнева. – 2006. – С. 15 – 17.
3. ПРОЛУБНИКОВ А.В. *Интервальный подход к решению задачи распознавания числовых матриц* // Вычислительные технологии. Том 17, №4. – 2012. – С. 77 – 87.
4. МОЛЧАНОВА В.С. *Адаптивный пороговый метод бинаризации растровых изображений технических чертежей* //

Прогрессивные информационные технологии. – 2015. – С. 62 – 70.

5. ТЕРЕХИН А.В. *Распознавание объектов методом вычисления оценок с использованием диагональных признаков формы* // Технические науки. Информатика, вычислительная техника. №1(29). – 2014. – С. 17 – 25.

VECTORIZATION OF RASTER IMAGES

Sergey Gusev, Institute of Control Sciences of RAS, Moscow, Ph.D. Student (*gs-serg@mail.ru*).

Abstract: This work is devoted to the development of tools for vectorization of color bitmaps. The aim of the work is to develop an application for the translation of color raster graphics into vector format. The paper describes the algorithm for vectorization of raster images Potrace, provides descriptions of vector formats Scalable Vector Graphics (SVG), Vector Markup Language (VML), Postscript, Adobe Portable Document Format (PDF), Adobe Illustrator.

Keywords: vector graphics, raster graphics, vectorization, format, algorithm, program.

Статья представлена к публикации
членом редакционной коллегии ...заполняется редактором…

Поступила в редакцию ...заполняется редактором…
Опубликована ...заполняется редактором…