Е. А. Булынко, Д. В. Коршикова, Е. В. Купчина

РЕАЛИЗАЦИЯ АЛГОРИТМА МАСШТАБИРОВАНИЯ КОПФА-ЛИЩИНСКИ НА PYTHON

Рассматривается алгоритм масштабирования Копфа-Лищински и его реализация на языке Python.

Введение

Растровая графика – это одна из форм представления цифрового изображения, состоящая из пикселей. Достоинства растровой графики в том, что она не требует большого количества ресурсов памяти, за счет использования палитры с узким цветовым спектром и малых размеров изображения. Одной из основных проблем растровой графики является сложность в изменении ее масштабов.

I. Алгоритм Копфа-Лищински

Данный алгоритм депикселизации, был представлен на конференции SIGGRAPH в 2011 году. Принцип алгоритма заключается в конвертации пикселей в полигоны с разной степенью сглаживания. B-сплайны – основные примитивы, определяющие сглаживание.

Сперва происходит преобразование пикселей в полигоны. Для получения сглаженных контуров квадратичные B-сплайны вписываются в последовательности видимых рёбер полигонов. Это создает плавность контуров, однако еще присутствует ступенчатый эффект. Поэтому необходима оптимизация контрольных точек B-сплайнов.Оптимизация происходит за счет поиска минимума суммы энергии каждого узла графа [1].

$$arg * min_{p_i} \sum_i E^{(i)} \quad (1)$$

Однако из-за низкого разрешения основной пиксельной сетки, контрольные точки сильно варьируются, результат может быть слегка ступенчатым. Поэтому на последнем этапе оптимизируются кривые линии, чтобы уменьшить ступенчатость. Изображение отрисовывается путём интерполяции цветов, с использованием радиальных базисных функций.

II. Реализация на Python

Для адаптации алгоритма под Python необходимо выделить основные этапы. Сначала происходит обработка вершин и ребер, после на основе этого создаются B-сплайны, затем происходит сглаживание готовой фигуры.

Для обработки вершин и ребер пиксельного изображения нужно создать классы class Point(object) и class Path(object). Их основная функция заключается в нахождении общих ребер соседствующих пикселей и общих вершин диагональных.

Класс class Shape(object) обрабатывает найденные вершины и ребра. На основе обработанных данных происходит смещение позиций вершин, результатом работы методов класса является набор точек, образующих новую, более гладкую фигуру.

Class BSpline(object) обрабатывает полученную угловатую фигуру, создавая вектор с точками, при этом пересчитывая их координаты. Заключительным этапом является общее сглаживание в классе SplineSmoother, путем создания новых точек и движение существующих.

Получившийся скрипт можно внедрять в открытый код некоторых графических фреймворков, но для создания отдельного приложения нужно прописать input и output изображений. На вход приложение будет получать пиксельное изображение, а на выходе отдавать сглаженное и адаптирующееся к масштабированию.

III. Выводы

Данный метод требует гораздо меньше ресурсов и делает обработку изображений в несколько раз быстрее. А Python весьма распространён и совместим с многими фреймворками, что дает широкую применяемость получившейся программе. Алгоритм сработается с современными мониторами, которые имеют большую кадровую частоту.

1. Снижко Е.А. Компьютерная геометрия и графика: Конспект лекций / Е.А. Снижко; Балт. гос. техн. ун-т. – СПб., 2005. – 132 с.
2. http://masters.donntu.org/2010/fknt/shekhovtsov/diss/index.htm

Булынко Екатерина Андреевна, студент 3 курса факультета информационных технологий и управления БГУИРа, ekatdrinagochkina@gmail.com.

Коршикова Дарья Валерьевна, ассистент кафедры вычислительных методов и программирования БГУИР, korshikova@bsuir.by.

Купчина Екатерина Валерьевна, инженер кафедры вычислительных методов и программирования БГУИР, e.kupchina@bsuir.by.

Научный руководитель: Кукин Дмитрий Петрович, заведующий кафедры вычислительных методов и программирования БГУИР, кандидат технических наук, доцент kukin@bsuir.by.