Задание 1. Фотоколлажи

Введение

Задача автоматического построения панорам в последнее время успешно решается методами компьютерного зрения. Появляются специализированные сервисы и приложения, например **photosynth** или **autostitch**. В этом задании вам будет предложено самим реализовать алгоритм автоматической сшивки панорам.

В задании предполагается, что изображения панорамы уже были выровнены друг относительно друга и нужно составить из этих изображений-лоскутков одну большую панораму. Задача решается в два этапа:

- Сначала в соответствии с каким-либо функционалом качества выбираются части изображения, которые попадут в итоговую панораму (сшивка)
- После составления панорамы происходит попытка выровнять разные её части, чтобы панорама визуально смотрелась целостной. Для этого используются методы блендинга, основанные на пирамидах изображений и смешивании изображений на разных частотах

Обязательная часть

Реализовать алгоритм сшивки и блендинга набора выровненных изображений одной панорамы.

Описание алгоритма:

- По набору изображений сформировать функционал, описанный в разделе "реализация". Узлами графа будут выступать пиксели панорамы. Нашей целью является присвоение метки каждому узлу графа таким образом, чтобы значение функционала было минимизировано
- После получения оптимальной разметки на финальном изображении нужно применить алгоритм блендинга, основанного на пирамидах лапласиан. Его лучше всего применять для каждого изображения-лоскутка панорамы в отдельности, смешивая его с его окружением на полученной панораме

Реализация

Алгоритм должен быть реализован в виде функции на Python со следующей сигнатурой *stitch_images(in_dir, mode)*, где *“in_dir”* – путь к директории с выровненными изображениями в формате *".png"*, *“mode”* – базовый (0) или бонусный вариант (1), возвращать функция должна сшитую панораму. Все изображения имеют один и тот же размер. Точки панорамы, которые данное изображение не покрывает, обозначаются черным цветом (0,0,0). Изображения частично перекрываются. Пример входных данных в файле *pano.zip*.

Для минимизации энергии при сшивке панорам необходимо воспользоваться реализацией алгоритма разреза графов maxflow. В ней для задания энергии необходимо сделать следующее:

- Изучить документацию maxflow
- Для каждой пары изображений-лоскутков посчитать унарные и парные потенциалы всех пикселей (пример использования в файле example.py)
- Создать несколько графов для каждой пары лоскутков размером N1xN2x2, где N1 и N2 – линейные размеры изображения. Все вершины должны быть связаны между собой, а также с двумя истоками (разметка ребер производится в соответствии с потенциалами пикселей)

- На основе метода разрезов графа понять, какой именно пиксель из двух картинок нужно вставить в конечное изображение.

Для сшивки панорам можно воспользоваться функционалом из статьи [2]:

$$C(L) = \sum_{p} C_d\big(p, L(p)\big) + \sum_{p,q} C_i(p, q, L(p), L(q)),$$

где $L(p)$ – значение энергии для разметки L, C_d – унарный потенциал, C_i – парный потенциал, p, q – граничащие пиксели, $L(p), L(q)$ – метки граничащих пикселей. Для правильной сшивки панорам нужно задать унарный штраф таким образом, чтобы черные области не попадали в итоговую панораму, если на их месте можно поместить значимую часть изображения-лоскутка. Парные потенциалы имеют следующий вид:

$$C_i\big(p, q, L(p), L(q)\big) = \big|S_{L(p)}(p) - S_{L(q)}(p)\big| + \big|S_{L(p)}(q) - S_{L(q)}(q)\big|,$$

где $S_L(p)$ – интенсивность изображения-лоскутка номер L в пикселе p. То есть данный парный потенциал предпочитает проводить разрезы там, где по крайней мере два изображения согласованы по интенсивности, что и обеспечивает плавные переходы.

Блендинг изображений-лоскутков на итоговой панораме нужно написать самостоятельно.

Бонусная часть

Добавить возможность задания с помощью мазков частей изображений, которые точно должны попадать в финальную панораму и применить эту технику к набору *family portrait* из [2], создав сшитое изображение семейного портрета. Мазки для каждого изображения должны задаваться с помощью двухцветных файлов, где белым цветом (255,255,255) задаётся мазок, а черным – всё остальное. Имена файлов с мазками должны иметь следующий вид – *'имя_файла_соответствующего_изображения_без_расширения_strokes.png'* и должны быть помещены в папку с изображениями-лоскутками.

Содержание архива с выполненным заданием

Архив должен содержать код и файл *readme*, со следующей информацией:

Общая информация:

- ФИО
- Система программирования, ОС
- Комментарии к реализации

Базовая часть:

- Склеенная панорама для набора *panoramic stitching* из [2]
- Сколько времени у вас заняло задание?
- С чем было сложнее всего разобраться?

Бонусная часть:

- Склеенная панорама для набора *family portrait* из [2] и файлы с мазками

Литература

1. **Image Mosaic**
2. **Interactive Digital Photomontage**
3. **Burt P. J., Adelson E. H. A multiresolution spline with application to image mosaics //ACM Transactions on Graphics (TOG). – 1983. – Т. 2. – №. 4. – С. 217-236.**

Данные

- Набор для сшивки панорамы – **pano.zip**
- Набор для создания семейного портрета – **family.zip**

Библиотеки

Необходимое библиотеки:

- Pymaxflow
- Cython
- Numpy

Рекомендую скачать:

- Matplotlib
- Scikit-image
- Scipy

Если какая-то из библиотек не встает, можно попытаться скачать уже готовую питоновскую сборку WinPython или Anaconda
Чтобы установить библиотеку на python, в командной строке необходимо ввести
$ pip install LIBRARY_NAME