

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ
по лабораторной работе №2
по дисциплине «Алгоритмы компьютерного зрения»
Тема: построение панорамного изображения

Студент гр. 2304

Ефремов М.А.

Преподаватель

Черниченко Д.А.

Санкт-Петербург
2017

Цель работы.

При помощи библиотеки OpenCV, используя предложенные изображения в качестве входных данных, получить панорамное изображение.

Основные теоретические положения.

На рисунке 1 показана реализация класса Stitcher библиотеки OpenCV, отвечающий за выполнение сшивающего конвейера. Используя этот класс можно управлять этапами выполнения конвейера.

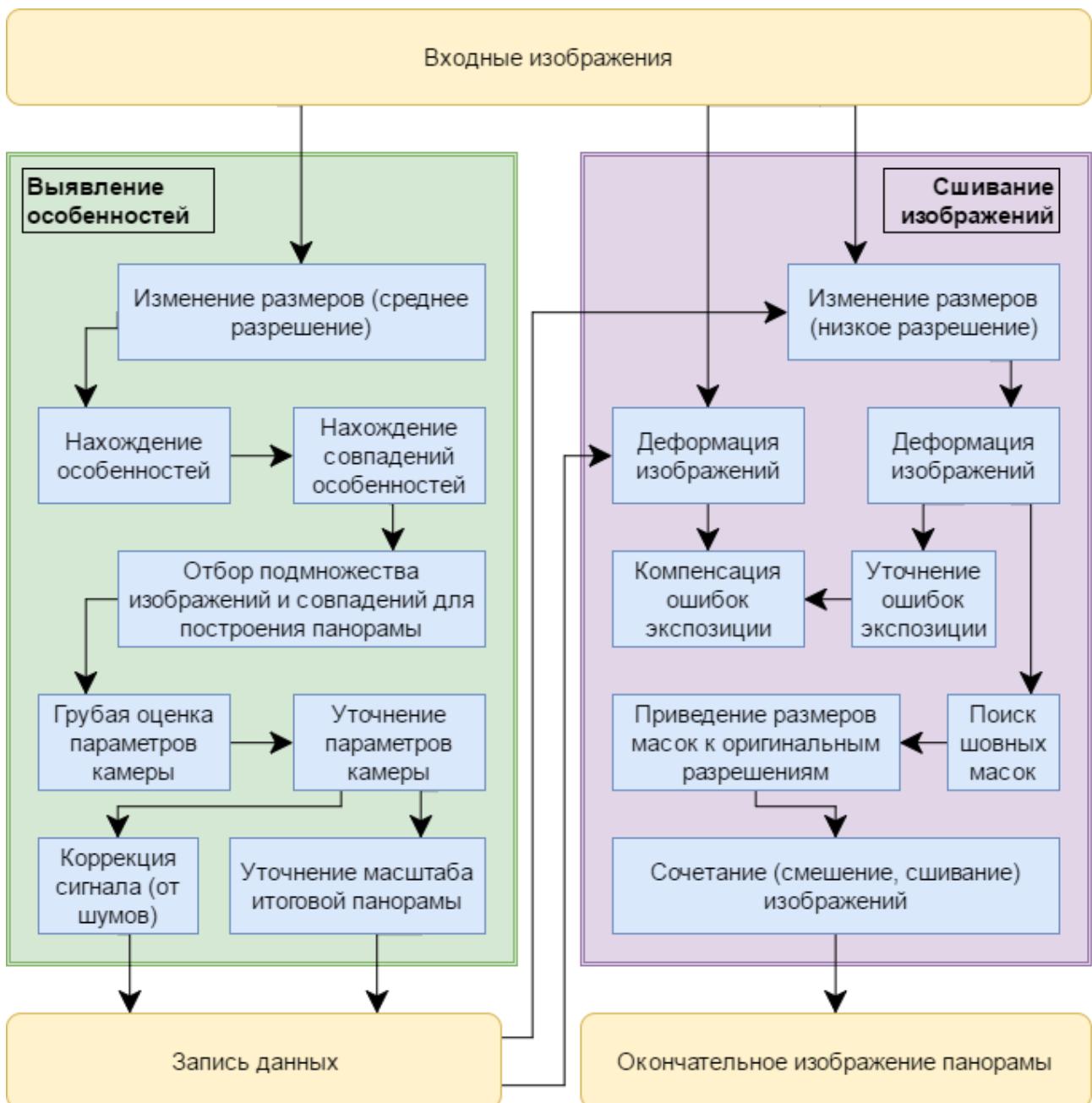


Рис. 1: схема работы сшивающего контейнера, реализованного в классе Stitcher.

Стоит отметить, что это общая последовательность работы класса Stitcher.

Данная схема была взята из документации к версии OpenCV 2.4 со ссылкой на статью: Brown and D. Lowe. Automatic Panoramic Image Stitching using Invariant Features. International Journal of Computer Vision, 74(1), страницы 59-73, 2007 года.

В OpenCV 3.2, использовавшемся при выполнении работы, конвейер сохраняется и для построения панорамы выполняются примерно следующие шаги:

1. Обнаружение ключевых точек (методы DoG, Harris, и т.д.) и описание локальных ориентиров (при помощи алгоритмов SIFT, SURF, и т.д.) по первым двум изображениям.
2. Нахождение совпадений описаний ориентиров между двумя изображениями.
3. Использование алгоритма RANSAC (RANdom SAmple Consensus — стабильный метод оценки параметров модели на основе случайных выборок) для оценки матрицы гомографии используя вектора описаний совпавших ориентиров.
4. Применение деформирующих преобразований, используя матрицу гомографии, найденную на шаге 3.

Основная разница между реализацией Stitcher в OpenCV 2.4 и 3.2 в методах обнаружения ключевых точек и описаний локальных ориентиров (таких как SIFT и SURF).

Программа

Далее на листинге 1 приведена программа, написанная для создания панорамы по двум данным изображениям.

```
1 #include <opencv2/imgcodecs.hpp>
2 #include <opencv2/stitching.hpp>
3
4 using namespace std;
5 using namespace cv;
6
7 int main()
8 {
9     String img_names[2];
10    img_names[0] = "images/left1.jpg";
```

```

11     img_names[1] = "images/right1.jpg";
12
13     vector<Mat> imgs;
14     for (int i = 0; i < 2; ++i)
15     {
16         Mat img = imread(img_names[i]);
17         imgs.push_back(img);
18     }
19
20     Mat pano;
21     Ptr<Stitcher> stitcher = Stitcher::create(Stitcher::PANORAMA, false);
22     stitcher->stitch(imgs, pano);
23
24     imwrite("stitching_result.jpg", pano);
25     return 0;
26 }
```

Листинг 1: Построение панорамы по двум изображениям

Экспериментальные результаты.

Для создания панорамы использовались фотографии, приведенные на рисунке 2, а результат работы написанной программы приведен на рисунке 3.



Рис. 2: Фотографии для создания панорамы



Рис. 3: Полученное изображение панорамы

Выводы.

На полученном изображении дублируется тумбочка, хотя шкаф оказался очень четким, как и календарь на стене. Во время выполнения работы, когда строилась панорама по четырем изображениям, использовался очень большой объем памяти (875 Мб по данным профайлера), в то время, как по двум не больше 400 Мб. Это объясняется тем, что из четырех фотографий по две были практически одинаковые и во время выполнения конвейера обнаруживалось множество совпадений, которые требовалось хранить в памяти.

Кроме того, сильно была заметна разница во времени выполнения конвейера.

Стоит отметить, что по неизвестной причине, исходные изображения не всегда хранятся так, как отображаются, например, просмотрщиком изображений. Так, исходные фотографии, при создании этого отчета, компонуясь системой L^AT_EX, были повернуты на 90° против часовой стрелки. В ходе тестирования исполняющего модуля на других машинах в 3 из 5 случаев панораму получить не удалось, скорее-всего из-за подобного странного хранения самих файлов фотографий.