

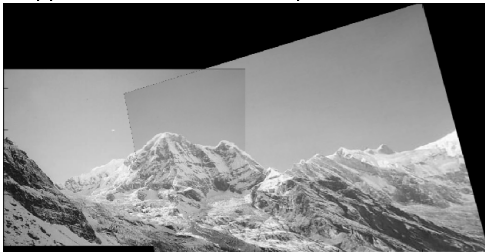
Семинар 7. Практикум

Задача

Создать скрипт автоматической склейки двух изображений в панораму. Необходимо будет склеить `demo1.jpg` и `demo2.jpg`.



Корректно выполненная реализация должна давать подобный результат:



Последовательность действий

Вычисление дескрипторов

Для обоих изображений вычислите SIFT дескрипторы (пользуясь библиотекой OpenCV) особых точек. Лучше взять дескрипторов побольше, так как это повысит надежность голосования в последующем RANSAC.

Сопоставление

Сопоставьте дескрипторы с помощью функции `match_descriptors` с параметром `cross_check=True` — функция осуществляет brute force сравнение и для каждого дескриптора первой картинки находит ближайший на второй. Если оба дескриптора являются ближайшими друг к другу, эта пара считается надежным матчем.

Визуализируйте сопоставления с помощью `plot_matches`. На этом этапе возможен высокий процент ложных сопоставлений.

Геометрическая валидация

Чтобы подавить ложные сопоставления, применяют геометрическую валидацию. Для этого находится такое проективное преобразование, которому удовлетворяет максимальное число матчей. Найдите такое преобразование с помощью функции `ransac` из `skimage.measure` с параметром `model_class=transform.ProjectiveTransform`. Минимальное число сэмплов для этого пункта — 4.

Вновь воспользуйтесь функцией `plot_matches` для визуализации матчей, прошедших валидацию. Качество матчей на этом этапе должно быть существенно выше, чем на предыдущем.

Визуализация панорамы

После того как получена матрица проективного преобразования, мы можем применить ее к одному из изображений, с помощью функции `warp` из `skimage.transform`.

Может оказаться так, что после преобразования часть изображения окажется за пределами области видимости. Чтобы этого не произошло, вычислите новые координаты уголков отображаемого изображения и увеличьте размер выходного изображения (параметр `output_shape` в функции `warp`), чтобы все уголки обоих изображений в нее попадали. Если часть изображения оказалась в области отрицательных координат, необходимо добавить преобразование сдвига `offset = SimilarityTransform(translation = [x, y])`, и в функции `warp` использовать суперпозицию преобразований полученных `ransac` и `offset`. Помните, что `offset` нужно применить к обеим картинкам!

Визуализируйте полученную панораму.