Image Processing and Pattern Recognition

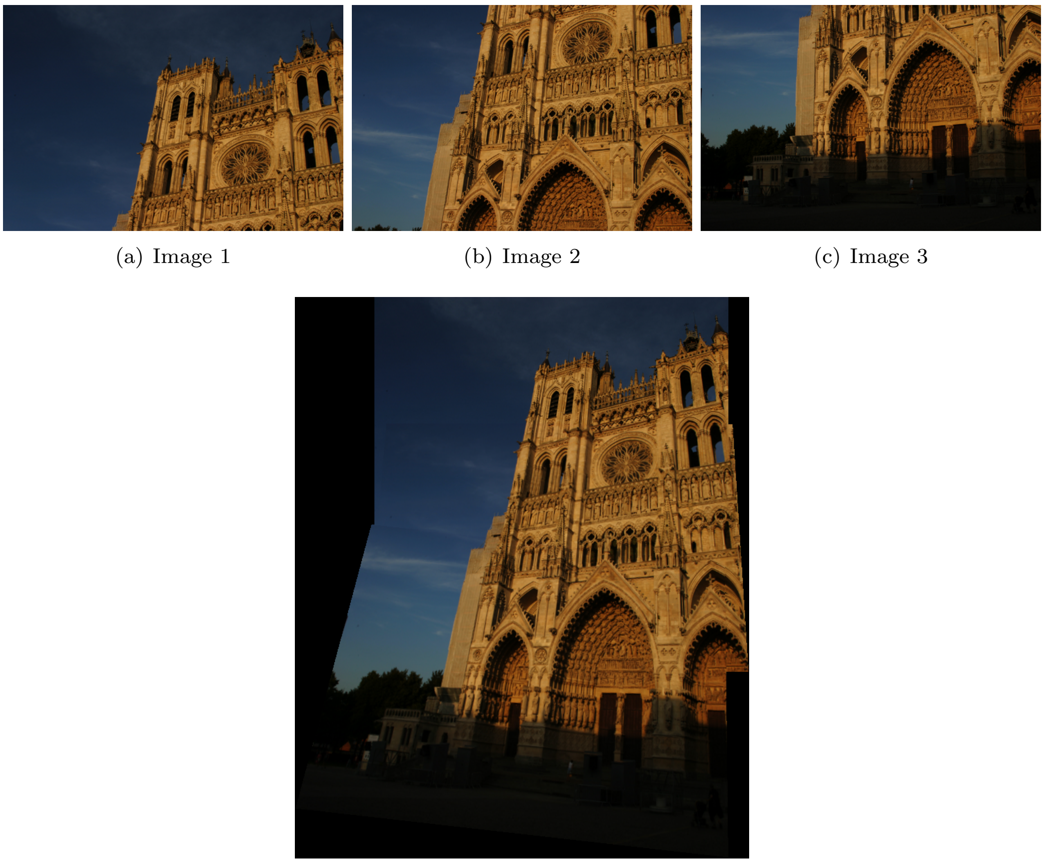
Assignment 1 - Image Stitching

October 17, 2016

Deadline: October 31, 2016 at 23:59h.
Submission: Upload report and implementation zipped (*[surname].zip*) to the TeachCenter using the "My Files" extension.

1 Goal

Image stitching is the process of combining two or more images that show different parts of the same scene into a single image. Modern cameras and smartphones often already include this functionality as part of their on-device processing.



(a) Image 1 (b) Image 2 (c) Image 3

(d) Stitched image

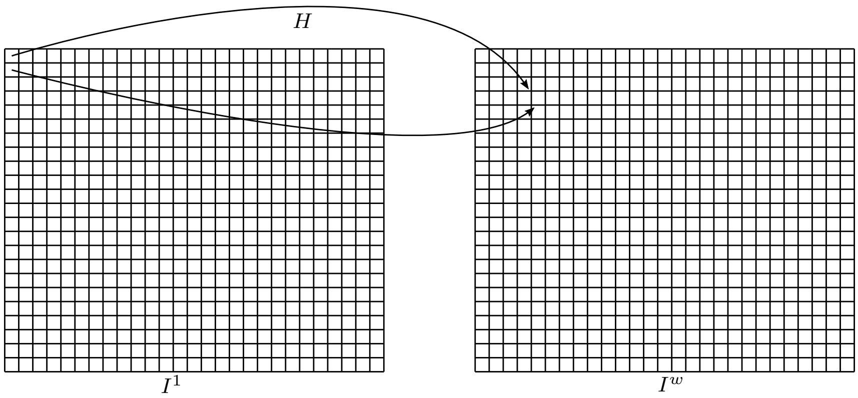
Figure 1: Image stitching

2 Method

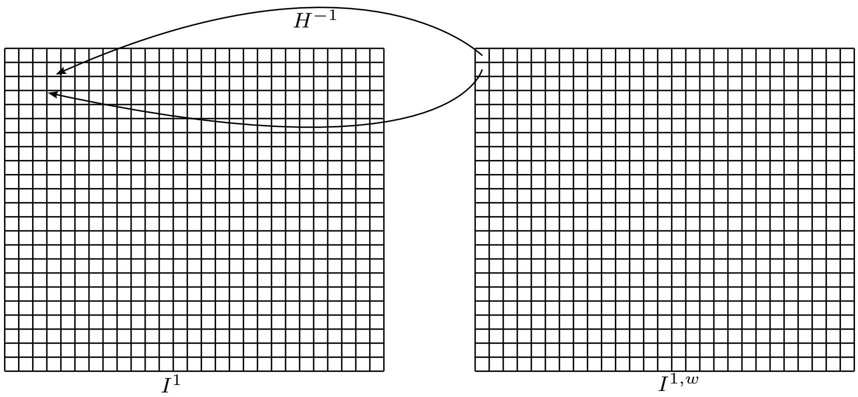
The process of image stitching involves finding homographies that map the images to a common reference plane. A homography is a projective transformation with 8 degrees of freedom (DOF) that can be represented as a matrix in $\mathbb{R}^{3\times 3}$. More specifically, a homography is a mapping of points from one plane to another plane.
Let $x = (x_1, x_2, x_3)^T \in \mathbb{P}^2(\mathbb{R})$ be a homogeneous point in the 2 dimensional projective space[1], a homography H maps the point x to a new point y according to

$$y = Hx \tag{1}$$

Note that the usual euclidean (inhomogeneous) coordinates can be obtained[2] from homogeneous coordinates by $x_{\text{inhom}} = \left(\frac{x_1}{x_3}, \frac{x_2}{x_3}\right)^T, \; x_3 \neq 0$.
The process of estimating homographies from the images is already implemented for you, your task is to compute the warped image (i.e. geometrically transform the image according to the homography) and assemble the final stitched image.



(a) Forward mapping



(b) Backward mapping

Figure 2: Forward (a) and backward (b) mapping. The forward mapping starts from pixel coordinates in I^1 and compute the corresponding coordinates in $I^{1,w}$. To assign grayvalues to the pixels in $I^{1,w}$, we have to round the computed coordinates. The backward mapping starts from pixel coordinates in $I^{1,w}$ and we compute corresponding coordinates in I^1. The grayvalue for a given pixel in $I^{1,w}$ is found by interpolating from the corresponding coordinate in I^1.

[1] elements of the space $\mathbb{P}^2(\mathbb{R})$ can be represented by homogeneous coordinates in $\mathbb{R}^3$
[2] except points at infinity which have $x_3 = 0$

2.1 Image Warping

Let us assume we have given a homography H_{12} that maps points from image I^1 to I^2 and we are interested in the warped image $I^{1,w}$. As usual, images are represented as matrices in $\mathbb{R}^{M\times N}$, where M denotes the image height and N is the image width. We refer to a pixel at index $(i, j),\ i = 1, \ldots, M,\ j = 1, \ldots, N$ using the notation $I_{i,j}$[3].

A naive approach to compute $I^{1,w}$ would be to go through all pixel coordinates $p = (j, i, 1)^T$ of I^1, compute the new position by $\tilde{p} = Hp$ and write the grayvalue of $I_{i,j}$ to the new position $\tilde{p}$. This is called *forward mapping* (see fig. 2(a)). However, due to the nature of the homography it frequently happens that the new position $\tilde{p}$ does not fall exactly onto the pixel raster. In order to write a value to the warped image we have to decide on a pixel position by e.g. rounding the coordinate $\tilde{p}$ to integer values. This leads to holes and artifacts, as some pixel coordinates of the warped image are never hit, whereas others are hit multiple times.

A better approach is to use a *backward mapping* (see fig. 2(b)): Here we go through all the pixel coordinates $\hat{p}$ of the new image $I^{1,w}$ and calculate the corresponding position in I^1 by $p = H^{-1}\hat{p}$. Using this approach we can find a grayvalue by interpolating from I^1 at position p. There are no holes or artifacts in the warped image, since we cover all possible pixel coordinates $\hat{p}$.

- Implement a backward mapping to compute the warped image
- Stitch the images together to obtain the final panorama

3 Framework

The framework already contains functionality to estimate homographies from images. Your task is to implement the warping and assemble a final panorama image with the given homographies. To interpolate grayvalues you can use the function `scipy.ndimage.map_coordinates()`.

[3]Note that here we use the usual matrix convention where i is the row index and j is the column index. On the other hand, when we think about coordinates we typically have the x-coordinate (horizontal) first, then y.