EE5175: Image Signal Processing

Lab-3

Image mosaicing

1 Problem statement

Image mosaicing is the alignment and stitching of a collection of images having overlapping regions into a single image. In this assignment, you have been given three images which were captured by panning the scene left to right. These images (`img1.png`, `img2.png` and `img3.png`) capture overlapping regions of the same scene from different viewpoints. The task is to determine the geometric transformations (homographies) between these images and stitch them into a single image.

2 Steps

1. Take `img2.png` as the reference image.
2. Determine homography H_{21} between $I_2 =$ `img2.png` and $I_1 =$ `img1.png` such that $I_1 = H_{21} I_2$.
3. Determine homography H_{23} between $I_2 =$ `img2.png` and $I_3 =$ `img3.png` such that $I_3 = H_{23} I_2$.
4. Create an empty canvas. For every pixel in the canvas, find corresponding points in I_1, I_2 and I_3 using H_{21}, identity matrix and H_{23} respectively (target-to-source mapping). Blend the three values by averaging them. Employ the values in blending only if it falls within the corresponding image bounds. Choose the origin so as to get a full mosaic.

2.1 Determining homography between two images

1. Determine SIFT features of the two images and determine correspondences between them. File `sift_corresp.m` returns the SIFT correspondences between two images (see Section 2.3). Now to find H such that:

$$\begin{bmatrix} x_i' \\ y_i' \\ 1 \end{bmatrix} \sim H \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

2. Run RANSAC on matched points (correspondences) to remove outliers (wrong matches), and find the homography between the two images.

(a) Input: Matched points (x_i, y_i) and (x'_i, y'_i) with $i \in \mathcal{M}$.

(b) Randomly pick four correspondences (so that we can form eight equations), i.e. (x_i, y_i) and (x'_i, y'_i) with $i \in \mathcal{R} \subset \mathcal{M}$ and $|\mathcal{R}| = 4$, where $|\cdot|$ denotes the cardinality of the set.

(c) Calculate the homography H using the above four correspondences (see Section 2.2).

(d) For each of the remaining correspondences (x_i, y_i) and (x'_i, y'_i) with $i \in \mathcal{P} = \mathcal{M} \backslash \mathcal{R}$, check whether they satisfy the homography (within an error bound). If yes, add the index of that correspondence to the consensus set.

$$\begin{bmatrix} x''_i \\ y''_i \\ z''_i \end{bmatrix} \leftarrow H \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}, \text{ and normalize so that } z''_i = 1,$$

$$\text{i.e. } x''_i \leftarrow x''_i / z''_i \text{ and } y''_i \leftarrow y''_i / z''_i$$

If $\sqrt{(x'_i - x''_i)^2 + (y'_i - y''_i)^2} < \epsilon (= 10)$, then update consensus set $\mathcal{C} \leftarrow \mathcal{C} \cup \{i\}$.

(e) If the consensus set is large enough i.e. if $|\mathcal{C}| > d (= 0.8|\mathcal{P}|)$, then return this homography H, else go to step (b).

(f) Output: Homography H.

2.2 Calculating homography

1. Consider a correspondence (x, y) and (x', y'),

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}.$$

Upon normalizing z',

$$\begin{aligned} x' &= h_1 x + h_2 y + h_3 / h_7 x + h_8 y + h_9, \\ y' &= h_4 x + h_5 y + h_6 / h_7 x + h_8 y + h_9. \end{aligned}$$

Form two equations for each correspondence (corresponding to two rows of matrix A).

$$\begin{aligned} (x)h_1 + (y)h_2 + (1)h_3 + (0)h_4 + (0)h_5 + (0)h_6 \\ - (x'x)h_7 - (x'y)h_8 - (x')h_9 = 0 \\ (0)h_1 + (0)h_2 + (0)h_3 + (x)h_4 + (y)h_5 + (1)h_6 \\ - (y'x)h_7 - (y'y)h_8 - (y')h_9 = 0 \end{aligned}$$

2. Solve the system,

$$A_{8\times 9}\begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_9 \end{bmatrix} = 0$$

   i.e., find the null space of A.

3. Homography matrix

$$H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix}.$$

2.3 Using SIFT

2.3.1 MATLAB

Files needed:

1. `sift_corresp.m`
2. `sift.m` and
3. `sift` (for GNU/Linux) or `siftWin32.exe` (for Windows).

Copy the above three files to the working directory. In GNU/Linux, check that the file `sift` has executable permission. If not, run `chmod +x sift` in a terminal.

Usage: `[corresp1, corresp2] = sift_corresp('img1.png','img2.png')`

2.3.2 Python

Files needed:

1. `sift.py`

Copy the above file to the working directory. Ensure that you have OpenCV-python (any version below 3.4.3) installed.

Usage:

```
from sift import sift as sift_corresp
[corresp1, corresp2] = sift_corresp(img1,img2)
```

2.4 Creating the mosaic

Pseudo-code (MATLAB style):

```
canvas = zeros(NumCanvasRows,NumCanvasColumns);
for jj = 1:NumCanvasColumns
  for ii = 1:NumCanvasRows

   i = ii - OffsetRow;
   j = jj - OffsetColumn;

   tmp = H21 * [i;j;1];
   i1 = tmp(1) / tmp(3);
   j1 = tmp(2) / tmp(3);

   tmp = H23 * [i;j;1];
   i3 = tmp(1) / tmp(3);
   j3 = tmp(2) / tmp(3);

   v1 = BilinearInterp(i1,j1,I1);
   v2 = BilinearInterp(i,j,I2);
   v3 = BilinearInterp(i3,j3,I3);

   canvas(ii,jj) = BlendValues(v1,v2,v3);

  end
end
```

where `(OffsetRow, OffsetColumn)` can be chosen such that the final mosaic will fit within the canvas.

3 Capturing your own data

Use your mobile phone camera to capture images (three or more), and run your code to generate the mosaic. Ensure that there is adequate overlap between successive images, and the camera is imaging a far-away scene (think why). Bring down the resolution of the input images such that the width < 1000 pixels, and convert them to grayscale before using them for mosaicing.