Multiple View Geometry

Lab2 – Feature Extraction and Image Registration

Lab Sessions: 3 (6 hours)
Total Workload: 15 hours
Programming Language: MATLAB

October 2024 (v2)

1. Objective

The objective of these two lab sessions is to also become familiar with SIFT and planar transformations for rigid image registration: how to extract invariant features and their descriptors, how to perform feature matching and how to use them to compute a homography. Most importantly, this should give you some feeling about the strengths and weaknesses of local feature-based approaches.

It should be noted that before you start this activity, you should read Lowe's paper on SIFT:

> *David G. Lowe,* ***"Distinctive image features from scale-invariant keypoints,"*** *International Journal of Computer Vision, 60, 2 (2004), pp. 91-110.*

The paper can be downloaded from here.

Students will have to:

1. **Test** SIFT features extraction on a set of underwater images.
2. **Register** pairs of images acquired at different locations, that show approximately the same area.
3. **Implement** different image motion models in the form of homography matrices and estimate these homographies from image pairs.
4. **Improve** the registration accuracy by means of data normalization and outlier rejection during the homography estimation.

2. Test data and auxiliary code

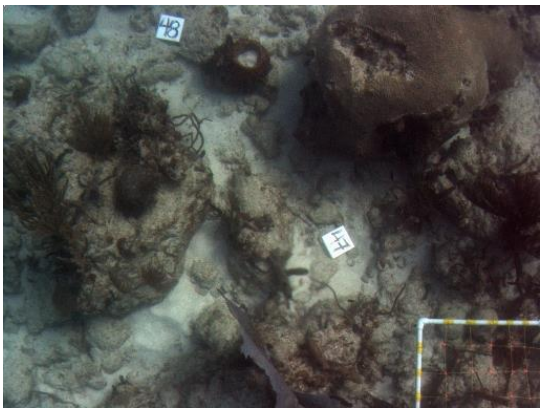
Introduction

The image set that we will use consists of pairs of underwater images that capture approximately the same area of the seafloor. The images were acquired over a coral reef patch, by a Remotely Operated Vehicle, in the coast of Florida.
By registering the images it becomes possible to estimate the motion of the vehicle or to building a photomosaic of the area.

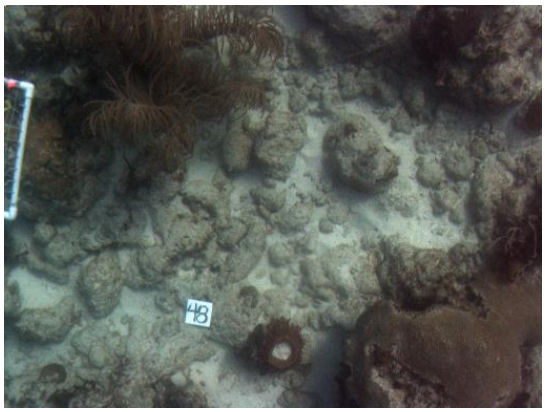
Step 1

Download and unpack the image datasets stored in the intranet of the course (file **MiamiSet00.zip**). and the auxiliary functions (file **matlab_helper_functions_lab2.zip**)

An example of one of the image pairs is shown in Figure 1.



(a) imgl01311.jpg



(b) imgl01396.jpg

Figure 1: **Examples of underwater images of a coral reef patch in south of Florida, USA, acquired with a ROV.**

Step 2

Use the script `lab2_section2_example.m` to visualize a pair of images and corresponding SIFT features.

Question 2.1

What is the meaning of the arrow's directions and lengths?

3. Feature Association

Introduction

Given a pair of images, we can detect a set of features and their associated feature descriptors. The descriptors can be associated according to their similarity, in what is known as **the feature association step**. This association will mainly rely on computing the distance between the descriptor vectors of each feature to determine the best candidate to represent the same feature in both images.

When this association has been done, the location of these feature pairs (or matchings) can be used to estimate a homography matrix. The homography matrix describes the geometric transformation of the coordinates of points between the two views, i.e. allows us to know where a given point in one image is on the other image. However this transformation is only valid if the two images are looking at flat area of the world.

During the matching process, some feature descriptors can be wrongly associated, due to several different factors that affect the appearance of the scene.

In this section we will see how the criteria for associating the descriptors can influence the quality of the matched descriptors.

Question 3.1

Check the function match.m provided by Lowe. What is the purpose of distance ratio (distRatio) and how is it used for the feature association?

Step 1

The function `matchsiftmodif.m` (with header below) is a modified version of match.m of Lowe. It extracts and associates SIFT features, and returns two list of coordinates (CL1uv and CL2uv) with the locations of the associated features.

```
[CL1uv,CL2uv] = matchsiftmodif(image1name, image2name, distRatio, drawMatches);
```

Test it for images imgl01311.jpg (as image1) and imgl01396.jpg (as image 2), and values of distRatio of 0.4, 0.6 and 0.8.

Questions 3.2

What is the effect of the parameter distRatio in terms of number of associations?
How can you tell visually that an association is wrong or not?
What is the effect of this parameter in terms of wrong associations?

For these two images we provide a precomputed homography H12, in the form of a 3x3 matrix which is defined up-to-scale. This homography, maps points from one image to the other (and vice versa), under the assumption that the scene is flat or approximately flat.

$$\lambda \begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix} = H12 \cdot \begin{bmatrix} u_2 \\ v_2 \\ 1 \end{bmatrix}$$

Step 2

Create a function called `projectionerrorvec.m` that takes, as input, two list of coordinates (CL1uv and CL2uv) on two images and the homography that relates them (H12). The function will return an error vector that contains the Euclidean distance between each point in CL1uv and its corresponding point in CL2uv after applying the homography H12.

```
errorVec = projectionerrorvec(H12,CL1uv,CL2uv)
```

Using this function, compute the error vector for increasing values of the distance ratio, where distRatio = 0.3:0.05:0.9 (that is, between 0.3 and 0.9 with jumps of 0.05).
Create a plot with the following indicators as a function of the distance ratio:

- Average error,
- Maximum error
- Number of associated features
- Number of associated features with error higher than a distance threshold of 50 pixels

For creating this plot use the images and homography described in the appendix (also available in the file H12.mat).

You can use the provided script `lab2_section3_empty.m` as a starting point for this, or write your own script entirely.

Questions 3.3

What conclusion can we draw from this plot?

- Is it possible to select a suitable value for the distance ratio for these images?
- Is it possible to select value for the distance ratio that guarantees no outliers, while providing a high percentage of associations?

4. Estimating a Homography to Register a Pair of Images

Step 1

Implement a Matlab function that, given two Nx2 matrices (CL1uv and CL2uv) containing a list of coordinates of features and correspondences ($[u_1, v_1]$ and $[u_2, v_2]$), and a string specifying a transformation model ('translation', 'similarity', 'affine' and 'projective'), will compute the homography matrix that describes the planar transformation. The homography matrix must be computed by solving the equation systems presented in the lecture notes. The implemented function should have the following header:

```
H12 = computeHomography(CL1uv,CL2uv, Model)
```

As a starting point you can use the provided empty function `computeHomographyEmpty.m` for this, or write your own function entirely.

Step 2

This step, based on synthetic data, will allow you to verify that the homography matrix has been correctly computed, for the different motion models.

Unpack the **DataSet01.zip** file available in the intranet of the course. Use the feature coordinates stored in the **Features.mat** file to compute the homographies that relate the reference image **00.png** with the other three, that is, **01.png**, **02.png** and **03.png**.

Identify the motion model that better fits the transformation of each image pair. Use the function `showwarpedimages(I1,I2,H12)` visualize the result of the registration.

Question 4.1

What are the motion models that better fit the planar transformation between the reference image and the rest? Visualize the registration results using overlay in different color channels (red for the reference image and green for the other, for instance).

Did you notice any benefit or drawback using motion models with more degrees of freedom than the theoretically needed in some image pairs? To evaluate the accuracy of the homography matrices computation, use the reprojection error as the error measure.

Step 3

Generate the lists of matched points for some of the images in **MiamiSet00.zip** (eg. 'imgl01311.jpg' and 'imgl01396.jpg', or any other image pair with overlap), using the `matchsiftmodif.m` function, then compute the corresponding homography and display the results using overlay.

Question 4.2

Does the obtained homography matrices allow registering the image pairs accurately, or are there significant misalignments?

5. Improving the Registration Accuracy

Introduction

As we saw above, some descriptors can be wrongly associated due to different factors affecting the appearance of the scene. These wrong associations, also known as outliers, will prevent an accurate computation of the homography matrix. To deal with this, we will need to use an outlier rejection strategy, such as RANSAC, to identify and remove these outliers.

Step 1

Improve the Matlab function that computes the homography matrix adding RANSAC outlier rejection to the pipeline.
The new function should also return the lists of inliers, and will have the following header:

```
[H12,INLIERS1uv,INLIERS2uv] = computeHomographyRANSAC(CL1uv,CL2uv,
Model)
```

Step 2

Compute the homography matrices for the previously tested image pairs, and compare the results before and after the use of RANSAC. Visualize the results using overlay and compute the corresponding reprojection error. State the number of matched features and the numbers of inliers found.

Step 3 (optional)

Aside from the outlier rejection, another strategy for improving the numerical stability and the accuracy of the homography estimation, is to normalize the data.

Implement a data normalization method to better condition the data before the homography computation. A small literature review (or web

browsing) might be required. Repeat again the previous experiment using the normalized data, computing also the reprojection error.

Question 5.1

Are the results obtained with the RANSAC-based homography estimation more accurate than the previously obtained? The use of data normalization helps reducing the reprojection error?

5. Deliverable

This Lab exercise must be done in groups of 2 students.

Your deliverable will consist of one report in PDF format, and one or more matlab .m files with your code. You will have to upload these files as a single ZIP file to Moodle before the deadline.

The deadline is one week after the last lab session of this exercise.

The PDF should include the names of both members of the group and:

- **Should detail how you have solved every step**
- Include the .m code that solve each step
- Include the obtained results per step (eg plots and figures with your added description and comments)
- **Discuss, when necessary, the results obtained**.

The Matlab code should be well commented. There is no need for extensive comments, but it should be clearly stated what each part of the code is doing.

Your Matlab code should run as is. You are encouraged to create your own functions to better structure the code. The main execution script should be clearly identified.

Appendix

Images names and corresponding homography for section 3

```
% Miami coral reef
image1filename = 'img101311.jpg';
image2filename = 'img101396.jpg';
H12 = [0.923963 0.105144 -41.64614; -0.105144 0.923963 -224.7121;0 0 1];


% Additional pair of La Lune shipwreck
image1filename = 'IMG_1253_small.JPG';
image2filename = 'IMG_1254_small.JPG';
H12 = [0.991966 0.135529 -85.4565; -0.135529 0.991966 -372.3004;0 0 1];
```