Mosaicing by SIFT

Lab Exercise 3 for Beeldverwerken

Informatics Institute
University of Amsterdam
The Netherlands

December 3, 2013

1 Introduction

Your task for this week is to write a report on the use of the SIFT to build an image mosaic. Instead of developing all code yourself, you may use existing libraries and built-in Matlab functionality.

Be aware that using software libraries written by others requires that you read the manuals carefully. Especially when a lot of math is involved it is important to get the details right. Things to look out for are:

- what is the coordinate system being used? Be aware that in Matlab different coordinate systems are in use.
- are the vectors given or required in homogeneous form or not?
- what is the shape of the matrices used? Be aware that sometimes Matlab requires the transpose of projective transformations (in relation to what we are used to).

Always read the documentation! Always test your code with simple examples.

And very importantly: never design your program from begin to end and only then start coding and testing. Design, code and test small parts and make sure it works before you move on.

First download and install the vlfeat library. This library can be found at `www.vlfeat.org`. Installation instructions can be found there as well.

The goal is to automatically make a mosaic out of two images. As an example consider the images `nachtwacht1.jpg` and `nachtwacht2.jpg` in Fig. **??**. You have to make a new image that transforms one image to the coordinate system of the other image (using a projectivity or straightening transform) and that combines the two images into a larger view of the nachtwacht.

2 Projectivity

We start by using Matlab built-in functionality for projective transforms. Make sure `nachtwacht1.jpg` and `nachtwacht2.jpg` are somewhere Matlab can find them (e.g. you might add the path to the Matlab path). And then run the script `demo_mosaic.m`.

Figure 1: Nachtwacht Images `nachtwacht1.jpg` and `nachtwacht2.jpg`.

You will see a figure showing the two images. You have to click at four corresponding points (the vertices of a quadrangle) in both images (in a clockwise fashion). Then the script calculates the projectivity from the point correspondences and shows the combined image as a result.

Study the code (in files `demo_mosaic.m` and `pickmatchingpoints.m`) and figure out what is done.

* **(1 points)** Replace the line: `T = maketform('projective',xy', xaya');` with your own code to calculate the projectivity matrix from possibly more than four points correspondences (using the svd 'trick'). We will need this later on!

3 SIFT

The goal is to find the point correspondences in an automated fashion. The scale invariant feature transform (SIFT) is very often used for this purpose. With the function call `[F,D] = vl_sift(image);` you can calculate the sift features in an image. With `vl_ubcmatch` you can match two descriptor sets to find possible matching points in two images.

But you will also need to read and understand the original paper by Lowe(2004) on the SIFT method. This is an important and much-quoted paper in Computer Vision. We want to show you that you can understand most of it by what you know now; and when and how to use other sources such as `wikipedia` to understand the rest of it. The documentation of `vl_feat` is also a very useful resource `http://www.vlfeat.org/api/sift_8h.html`, clarifying the structure in an excellent example of documentation of scientific software.

Theory Questions

3.1. First read "How to Read a Paper", Appendix B.1.1. Then read the Lowe paper through once, to understand its structure. It is a big paper, so use your common sense to see at what level of detail to read to ixdentify the crucial sections. (This paper is actually not that well structured by the author; critique him in your mind and learn from this example.) *At the very least, we expect this to have been done on Monday before class (!) – but please do not stop there.*

3.2. Now we understand what the framework is for. Figure 12 is great! Relate SIFT in you mind to the mosaicing task, so you can identify which parts of the paper are important to understand.

3.3. **(3 points)** You should have seen that there is quite a lot of theory that we already know. But the term *'scale space'* is new. Look it up on wikipedia. Also find out what 'scale space extrema' are, and what they mean intuitively.

3.4. **(3 points)** Scale space extrema are going to be detected by computing D according to (1). According to Lowe "the relationship between D and $\sigma^2\nabla^2 G$ can be understood ..." but he does not return to D. Close the loop in his explanation yourself.

3.5. **(3 points)** If you did not look it up before, ∇^2 is the Laplacian, in 2D images it can be computed as the trace of the Hessian, i.e., $\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$. Prove that the trace of the Hessian is equal to the sum of the eigenvalues (see page 12, where he uses this fact). Hint: use the eigenvalue decomposition on the Hessian. Is it always possible to diagonalize the Hessian?

3.6. **(1 points)** Lowe interweaves implementation considerations with theoretical stuctural explanations. I find that annoying, but at least we see his rationale towards choosing his parameters. Find all the magic numbers in his implementation, and signal which you may have to vary for your mosaicing task.

3.7. Section 4: fitting a local quadratic function (I think he means a 2nd order polynomial): consult the Facet Model in Chapter 2.7 of the Lecture Notes (or the 2011 eerste deeltentamen). It is a straightforward application of our linear algebra tools, and you see that this is not explained in detail: we are supposed to know.

3.8. **(4 points)** Make sure that you understand the equation on page 11: compared to (2) on page 10, the 1/2 looks like a typo. Is it?

3.9. **(2 points)** Page 15, line -4, mentions trilinear interpolation. Look it up in `wiki`. Why are we in 3D, what are the meanings of the dimensions of the space we are in? (Hint: it is not scale space!)

3.10. **(1 points)** Page 16, line 12: what does he mean by "affine changes in illumination"?

3.11. Section 7.3: the Hough transform will be treated next week. The key idea is 'clustering by voting' as they mention in the second paragraph. In the lab assignment, we will use RANSAC instead.

* **(15 points)** In your report you should describe the SIFT in enough detail for your fellow student (who took the vision course but was not there when SIFT was discussed) to be able to understand it.

* **(5 points)** Explain in your report why SIFT is scale and rotation invariant.

4 RANSAC

The possible matching points resulting from the `vl_ubcmatch` function will contain a lot of wrong matches. Therefore we have to decide which matching points are needed in the estimation of the perspectivity. This is where the RANSAC algorithm comes in handy.

* **(15 points)** Read the explanation of the RANSAC algorithm on Wikipedia and code it in Matlab to be able to estimate a projectivity.

* **(2 points)** In your report you must describe how many iterations are needed to be reasonably sure to find an optimal solution.

5 Report

Your report should contain the following sections:

- **(2 points) Introduction**
- **Theory.** In this section you describe the theory on which your program is based.
  - **(5 points) Perspectivities and perspectivity estimation.** For this section a short overview of the relevant equations and definitions is enough.
  - **SIFT.** (see in a previous section what is needed in this section, and how many points are awarded).
  - **(5 points) RANSAC.** (see a previous section)
- **(15 points) Algorithm.** A description with pseudo code, or well documented Matlab, explaining the entire mosaic construction.
- **(7 points) Experiments.** The results of applying your code to the nachtwacht and possibly other pictures.
- **(2 points) Conclusions**

The total length of the paper should not exceed 6 pages (excluding code in the appendix).
MAX POINTS: 91