Universitat Politècnica de Catalunya
Universitat de Barcelona
Universitat Rovira i Virgili

Master in Artificial Intelligence

Computational vision

Feature detection and matching (II)

Authors:
Alejandro Suárez Hernández
Johannes Heidecke

October 2016

Contents

List of Figures

1 Feature matching

1.1 Question 1 (on increasing the threshold ratio)

Modify the previous threshold. What is the effect of this threshold? Comment your response.

We can see the result of the default matching process in figure 1.

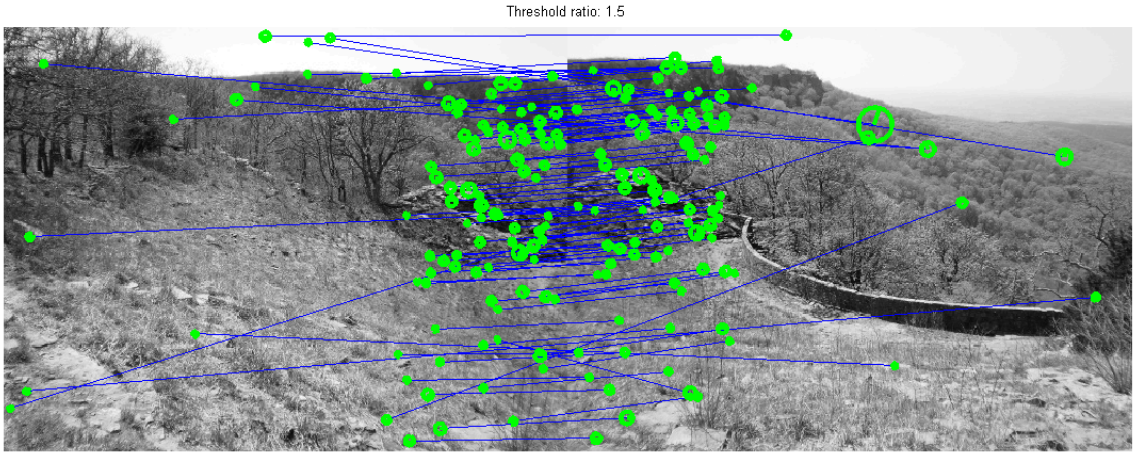

Figure 1: Result of matching with a threshold ratio of 1.5 (default)

We have also computed the matches using different thresholds. In figure 2 we can see the result of the matching algorithm with a threshold ratio of 2.0. It is clear that there are fewer matches. This is due to the fact that the ratio between the distances to the first and second best matches acts as an uniqueness measure. Therefore increasing the threshold leads to a more restrictive selection of the relevant matches and to a lower number of matches at the end. In other words, since we consider just the matches that accomplish $\frac{d_2}{d_1} \geq THR$ (being d_1 and d_2 the distances to the closest and second closest correspondences), a higher value of THR filters out matches with a ratio $\frac{d_2}{d_1}$ closer to 1.

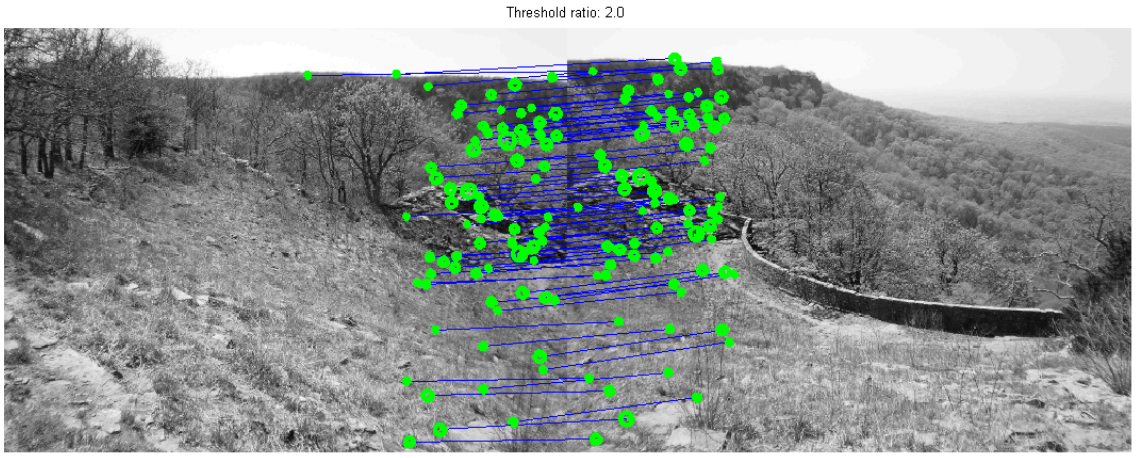

Figure 2: Result of matching with a threshold ratio of 2.0

On the other hand the effect is reverted when we reduce the threshold. In figure 3 we can see that there is a very notable increment in the number of matches when we set the threshold to 1.0.

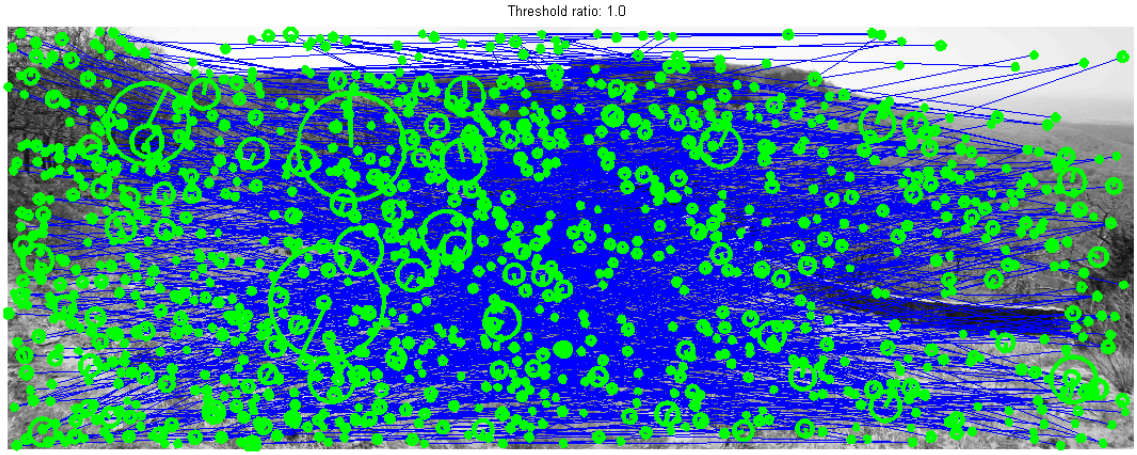

Figure 3: Result of matching with a threshold ratio of 1.0

1.2 Question 2 (on introducing a linear model)

Comment on the obtained result. Is the resulting model (more or less) correct? By looking at the original images, do you think a linear model is enough to model the transformation between the two images?

Figures 4, 5 and 4 show the result of combining with a simple translation model the matching results obtained with thresholds 15, 20 and 10, respectively.

Clearly, the best result is obtained in figure 5 (THR=2.0), thanks to the fact that the greater threshold manages to filter out the worst matches. On the other hand in figure 4 (THR=1.5) we can see the harmful effect of these bad matches in the overall result. This is even more evident in figure 6 (THR=1.0).

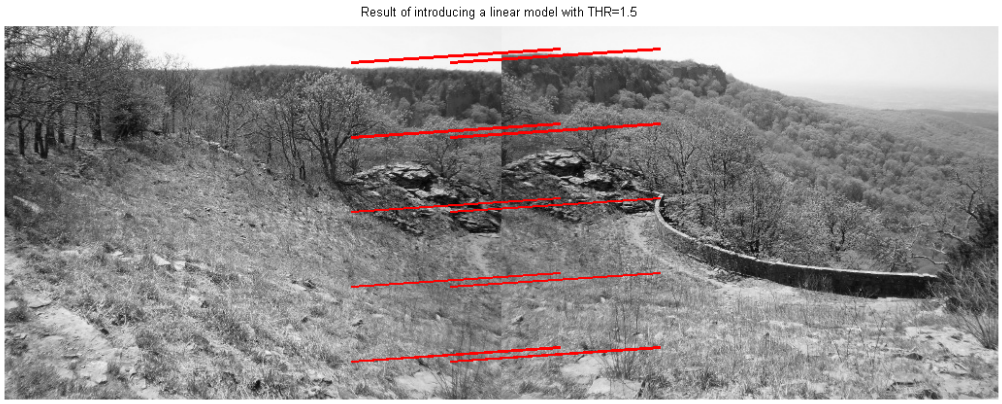

Figure 4: Result of matching with THR=1.5 combined with linear model.

All things considered, the simple translational model seems to perform nicely for the particular case of these two images as shown by the best of these three matches. This is due to the fact that the difference between the two images can be seen as a camera pan without significant rotation or skew. Therefore, the model yields to good results.

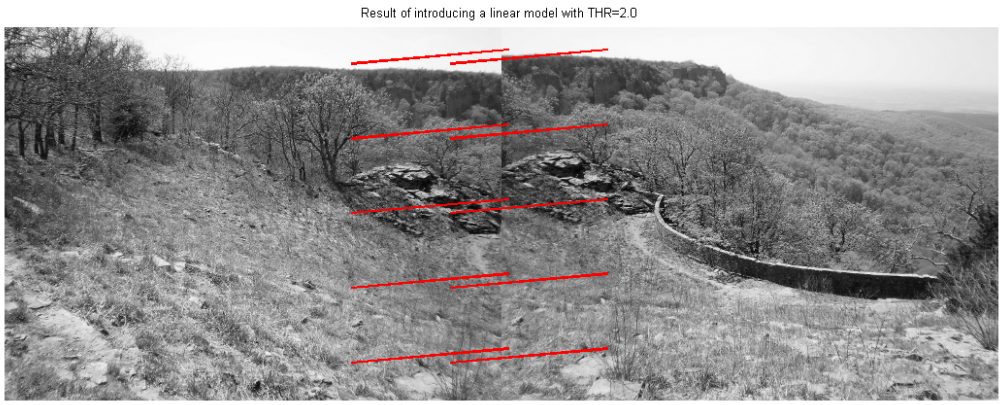

Figure 5: Result of matching with THR=2.0 combined with linear model.

Figure 6: Result of matching with THR=1.0 combined with linear model.

1.3 On the introduction of an afine model

Figures 7, 8 and 9 show the result of combining with an afine model the matching results obtained with thresholds 15, 20 and 10, respectively.

Much like in the previous case, and for the same causes, the best result is obtained with THR=2.0 (figure 8). However, while good, the result is not much better than the (already good) results obtained in 5. On the other hand, the matches showed in figures 7 and 9 are actually worse than those showed in figures 4 and 6. The reason behind this has already been posed in the assignment's statement: the increase in the number of the model's DoF does not contribute to a better match when there are many spurious correspondences. To this we add that neither do these additional DoF contribute meaningfully when a good match can be found with the adequate threshold and a simpler model.

1.4 Using a subset of with just the best matches

Now we will analyze what happens when we select only the a subset with the best matches to fit a model. From now on, we will deal exclusively with the matches found with THR=1.5 since the following technique aims at improving the fitting of the model even if an excess of matches has been found due to a low threshold.

In figure 10 we can observe a very clear improvement over the computed model in figure 4. When we select only the best matches, we are discarding the most spurious one. All

Figure 7: Result of matching with THR=1.5 combined with afine model.

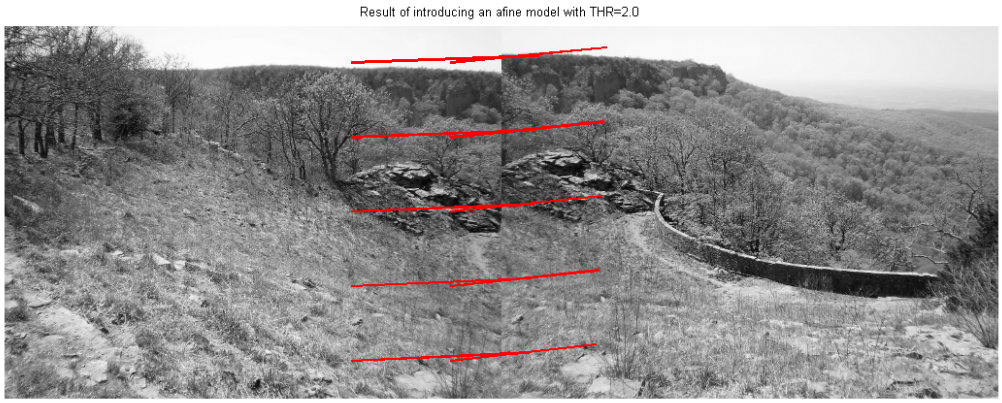

Figure 8: Result of matching with THR=2.0 combined with afine model.

in all this leads to a much better result, comparable to that of figure 5 (linear model after performing matching with THR=2.0). Therefore, this technique help us to obtain good results even when we cannot fine tune the THR parameter.

In figure 11 we can observe the result of applying this same technique with the afine model. We can see that the result is worse than before. As the assignment's statement asserts, the least means square method is very sensible to outliers when applied to the afine model. This can be seen when we reduce the cardinality of the subset that we use to compute the model from 10 to 9. The result can be seen in figure 12. This suggests that the 10th "best" match is actually an outlier which happens to have a low distance between descriptors.

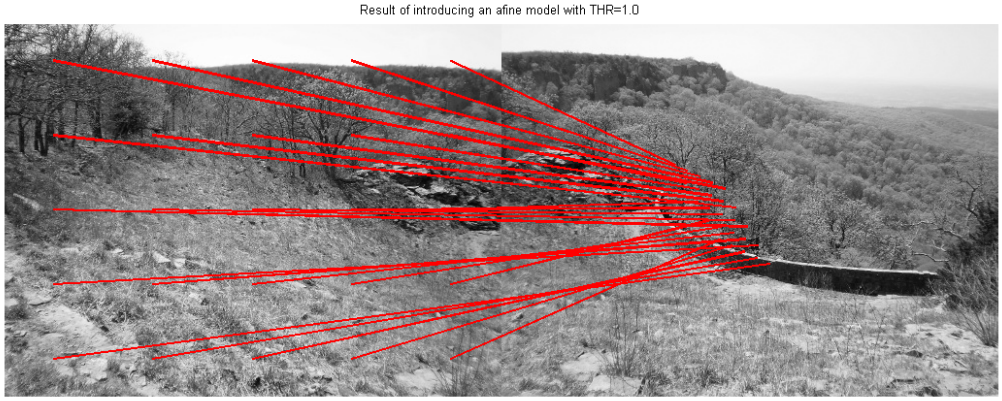

Figure 9: Result of matching with THR=1.0 combined with afine model.

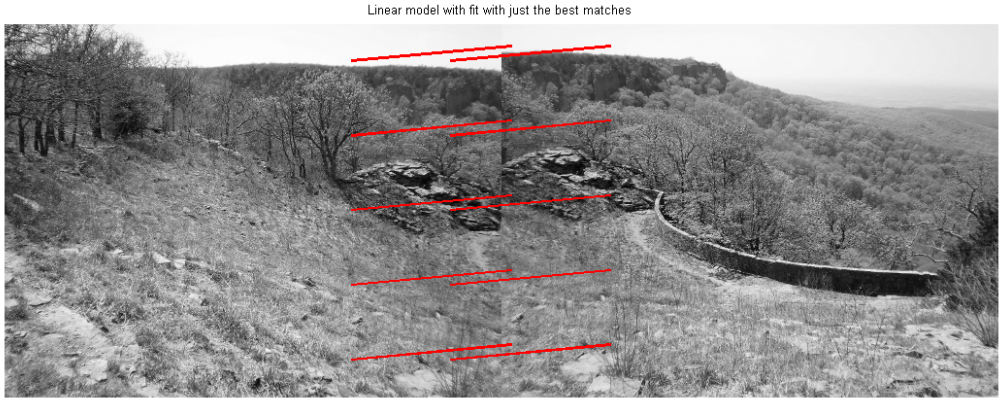

Figure 10: Result of linear model using just the 10 best matches

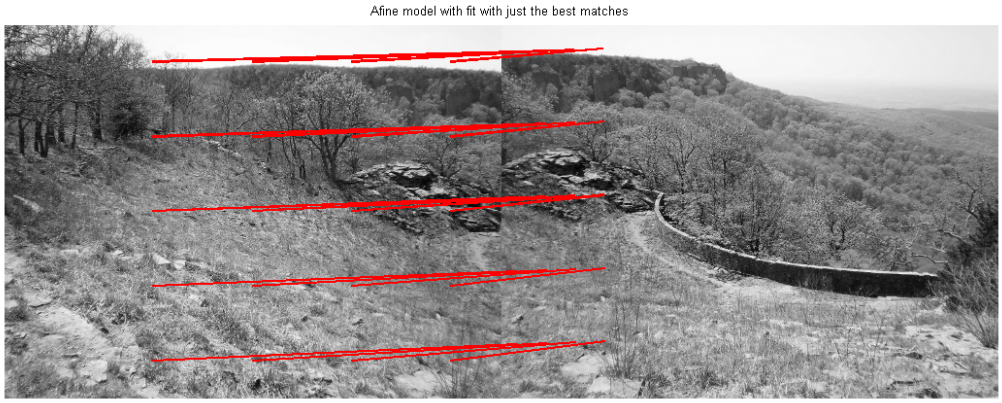

Figure 11: Result of afine model using just the 10 best matches

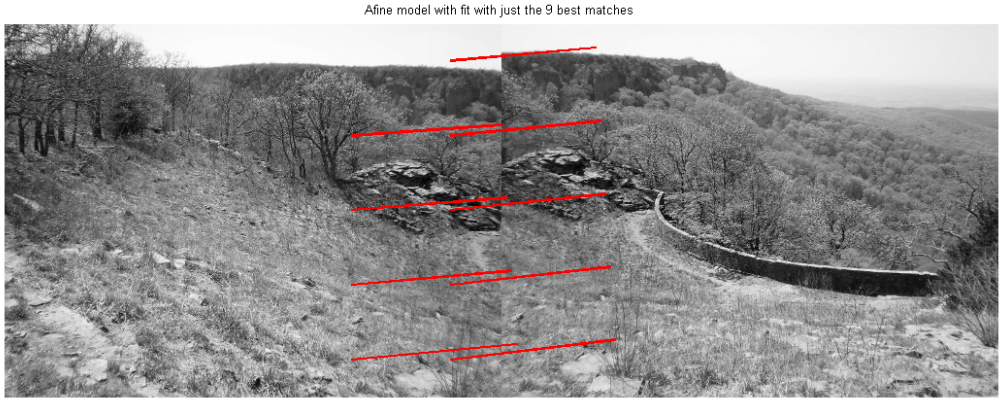

Figure 12: Result of afine model using just the 9 best matches

2 Panorama creation

We have implemented a method that performs the requested functionality in file `panorama_creation.m`.

Figures 13, 14, 15 and 16 show the panoramic images we have obtained with the first, second an third provided examples. The fourth panoramic corresponds to a custom examples provided by us (images `under_the_sea_1.png` and `under_the_sea_2.png` in the `images/` folder.

Figure 13: Panoramic created from first example

Figure 14: Panoramic created from second example

We can see that the results in all cases are quite acceptable. The second example is the one with the most perceptible transition from one half of the image to the other due to small changes in the brightness, the misalignment of the clouds and the non-still waters. In the rest of the cases, the transition is smooth enough to be overlooked by all but the eagle-eyed viewer.

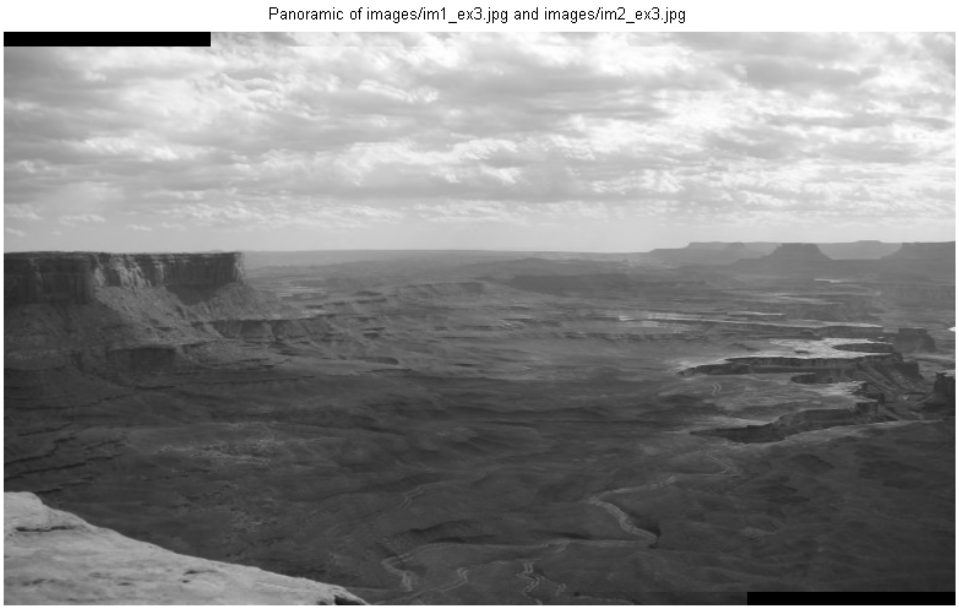

Figure 15: Panoramic created from third example

Figure 16: Panoramic created from fourth example (custom example)

Appendices

A Annex

Here we list all the delivered script and function files delivered for this practice, aside from those that were already provided to us. We include relevant observations. For full insight, we refer the reader to the source code.

- `feature_matching.m`: code for the first section (Feature Matching) of the assignment. The script is divided into several cells so they can be tested more conveniently.
- `panorama_creation.m`: method that performs the functionality requested in the second section of the assignment (Panorama creation). The method accepts two image paths and three optional parameters: `M`, the number of RANDSAC iterations (default 100); `N`, the cardinality of the subset built at each RANDSAC iteration (default 10); and `epsilon`, how close has to be a point from the second image to its correspondence in the first image in order for it to be counted as a "success" by the RANDSAC algorithm (default 5 px).