Modern to Historical Image Feature Matching

Robert Wolfe*
Supervised by Professor Anthony Whitehead, School of Computer Science

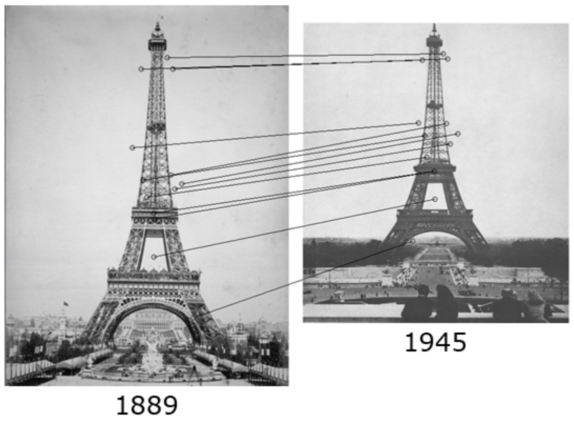


Figure 1: *An example of feature matching between two images taken at different time periods.*

Abstract

Matching images with each other using their unique features has many applications. While there currently exists many robust image feature matching systems, we propose that they are currently only accurate when matching modern photos with each other. Historical photos, traditionally taken on film, when converted to digital format are not likely to successfully match with modern equivalents. Photos of famous landmarks from various dates were gathered to verify this hypothesis. Images were matched using four standard feature matching systems. These systems matched either image corners or image blobs with each other. Match successes were recorded for each image comparison, and analyzed in tables. The results showed that the feature systems accurately matched modern photos with each other. However, when matching historical photos to modern photos, results were poor. A novel approach to feature matching using image regions was proposed. The expectation was that using regions instead of corners or blobs would result in higher accuracy when matching. Initial results were mixed, and no concrete solution was found.

Keywords: image matching, feature matching, computer vision

*e-mail:contact@robbiewolfe.ca

Acknowledgements

I would like to thank Professor Whitehead for supervising this project, clarifying issues, providing feedback and helping to write this report.

1 Problem Description and Domain

Image feature matching is an active research area for Computer Vision. The ability to match objects of interest across multiple images has many applications. There currently exists numerous techniques suitable for this task, each with their own pros and cons. The problem with modern techniques is they frequently rely on the information embedded in digital images to perform their feature matching. This poses a problem for historical photos, which have to be converted to digital format, usually through a scanning process. This conversion lacks the information which would be inherent in images taken with a digital camera. Our theory is that feature matching historical images with modern images will be significantly less accurate than desired. A new technique needs to be developed which accounts for the lack of digital information in the historical photos.

2 Motivation

With the rise of interest in Augmented Reality technology, the need for accurate historical to modern image matching is increasing. An example application could be that a user would use a camera phone on a famous landmark and the application would superimpose a historical image of the same landmark on the screen [Whitehead 2013]. Another could be the user uses a camera phone on an old photo of a famous landmark, and the application would match it to modern photos of the same landmark. Applications like these would be unusable if the matching was inaccurate.

3 Project Goals

The goals of this project are as follows: To demonstrate that modern feature matching techniques are inaccurate at matching historical to modern photos. This will be done by testing each technique on 10 famous landmarks, and recording match successes and failures. These results are to be displayed in tables, which will demonstrate each technique's accuracies. As well, the goal is to propose a method for more accurate matching.

4 Feature System Descriptions

Each feature system computes two components used for image matching: keypoints, which are the locations of desired features, and descriptors, which are collections of information on the pixels surrounding each keypoint. Descriptors are stored in a matrix, where each row of the matrix is the descriptor for a particular keypoint. The following is a brief description of each of the feature systems tested in this project. It will describe what they each consider a desired feature, how these features are discovered, as well as how the descriptors are gathered.

4.1 HARRIS System

The Harris feature system is a corner detector [Harris and Stephens 1988]. In this context, corner means a feature point with large intensity changes in more than one direction [Roth 2012]. The intensity change at a given pixel in a given direction vector can be computed using the sum of differences squared of the pixel values in the pixel of interest's neighbourhood and the associated pixel shifted by the vector. This sum can be expressed as the second moment matrix [Bay et al. 2008]. This matrix's determinant and trace can then be used to determine the corner response of a given pixel [Harris and Stephens 1988]. If the corner response is greater than a given value, that pixel is determined to be a detected corner, and a keypoint is stored.

The Harris descriptor for a keypoint is the pixel intensity values of the pixels in its neighbourhood. For this project, a neighbourhood was 7X7 in size, thus three pixels in every direction from the pixel. The intensity values are stored in row-major order.

4.2 KLT System

The Kanade-Lucas-Tomasi feature system (KLT) is another corner detector. Its feature extractor was originally described in a paper by Kanade and Lucas, and later expanded into a technique called ”Good Features to Track” [Shi and Tomasi 1994]. It was originally developed to track features in an animated movie, only keeping features that are not too dissimilar between frames. The OpenCV implementation of ”GoodFeaturesToTrack” was used in this project. The features are computed using the same matrix as HARRIS, except it uses the eigenvalues of the matrix instead of its determinant and trace [Bouguet 2000].

The matrix and its eigenvalues are computed over every pixel in the image. The minimum eigenvalue is retained for each pixel. Subsequently the maximum of these eigenvalues is retrieved. All pixels that have a minimum eigenvalue larger than a percentage of the maximum are then retained. From those pixels, only those whose minimum eigenvalue is larger than any other pixel in its 3x3 neighbourhood are kept. The retained pixels are further reduced by only keeping pixels who are a given minimum distance from any other retained pixel. These remaining pixels are the final keypoints. The KLT descriptor is the same as the one used by HARRIS.

4.3 SIFT System

The Scale Invariant Feature Transform (SIFT) feature system was originally developed as a means to discover image features that are invariant to scale and rotation [Lowe 2004]. This is accomplished using a cascaded approach, where computationally expensive operations are executed only on regions that pass certain initial tests. First, scale-space extrema are detected. This is accomplished by using a Difference of Gaussian function to identify potential interest points across all Gaussian scales and image locations. Local extrema are detected by comparing each sample point to its 8 neighbours in the current image and 9 neighbours in the Gaussian scales above and below it. The sample point is selected only if it is larger or smaller than all of these neighbours. Keypoints are then generated from the candidate locations and are computed as vectors, storing their scale, orientation and location. The orientation of each keypoint is based on the image's local gradient directions. Keypoints are rejected if they have low contrast or are poorly localized along an edge.

Transforming the image data by the scale, orientation and location information stored in each keypoint provides invariance to these parameters. Unlike HARRIS and KLT, SIFT descriptors are computed using the gradient magnitude and orientation at each sample point in a region around the keypoint location. These samples are then accumulated into orientation histograms, which are finally stored in a vector, representing the descriptor.

4.4 SURF System

The Speeded Up Robust Features (SURF) feature system can be classified as a blob detector [Mikolajczyk and Tuytelaars 2008]. It is a scale invariant feature detector based on an approximation of the Hessian-matrix. Rather than calculate the second order Gaussian derivatives in the Hessian-matrix for each pixel location, they are approximated in constant time using a pre-computed ”integral image”. These integral images provide a way for the sum of pixel intensities in a box area to be calculated in only 4 constant time operations, regardless of the size of the area. The determinant of this approximated matrix represents the blob response in the image at a specific pixel location. These blob responses are calculated for each location at multiple scales in constant time using the integral images [Bay et al. 2008]. Keypoints are kept where this determinant is maximum. Using multiple scales makes this feature system scale invariant.

The descriptor for SURF is similar to the one used by SIFT, since it focuses on the spatial distribution of gradient information. To speed up descriptor computation, instead of using gradients, integral images are used to compute the responses in the x or y direction at any scale. For each keypoint, multiple responses are computed using the interest point as its center. The sum of these responses represents an estimate of the dominant orientation of a pixel and its neighbourhood. The descriptor is constructed using a square region centered around the keypoint and oriented along the dominant orientation. It consists of 64 values, representing the intensity structure of each of the 4x4 sub-regions in the constructed square region.

5 Data Set Description

The feature matching tests were evaluated on 10 famous landmarks. Choosing famous landmarks made it likely to find on the internet numerous images from a large range of time to test on. Each image of the landmark would need to be from a similar point of view, otherwise the matching would not work, regardless of the time span. Each landmark had 10 images gathered from various dates. An example of a historical photo and a modern photo can be seen in Figure 2. The year the photo was taken was recorded. The images were also cropped and resized if necessary, so that they were all similar. All the images were gathered from Flickr or Google images. A summary of the landmarks used for testing is displayed in Figure 3. Each landmark displays its name, the range of dates of their respective photos, and 5 examples of the photos used for testing, in ascending order by date taken.

Figure 2: *Example of Historical (1858) and Modern (2006) photos of the Egyptian Pyramids used for feature matching.*

6 Tests on Data Set with Existing Feature Systems

All the images were preloaded in greyscale mode. The landmarks were tested against each of the 4 feature systems. For each system, each image in the landmark was compared with all the other images, plus itself. This resulted in 10 landmarks X 10 images X 10 images X 4 feature systems = 4000 comparisons. The keypoints and descriptors for the images being compared were computed using the particular feature system. Figure 4 shows an example of two images being compared using SURF, with their respective keypoints displayed. A naive matching of keypoints on the left to the keypoints on the right would result in too many match lines, as shown in Figure 5. To evaluate if a match was successful or not, a significant number of the match lines had to be discarded, retaining only matches deemed appropriate.

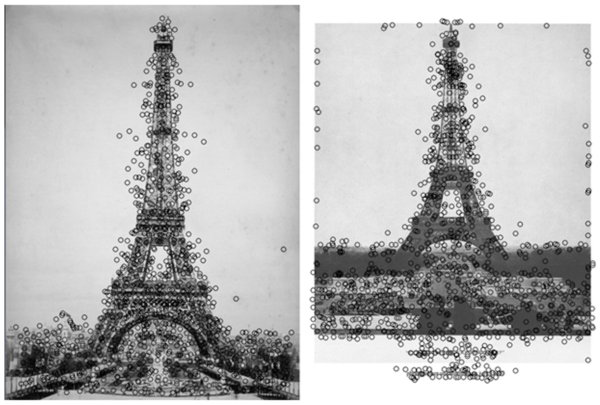


Figure 4: *Two images of the Eiffel tower being compared; showing their respective SURF keypoints*

The first step of matching keypoints between two images was to use the Fast Library for Approximate Nearest Neighbours (FLANN) search algorithm. FLANN takes a data set and a desired precision as input and executes what it determines to be the fastest approximate nearest neighbours algorithm to use, and returns the data that matches best [Muja and Lowe 2009]. Figure 6 demonstrates that FLANN greatly reduces the number of matches, making it easier

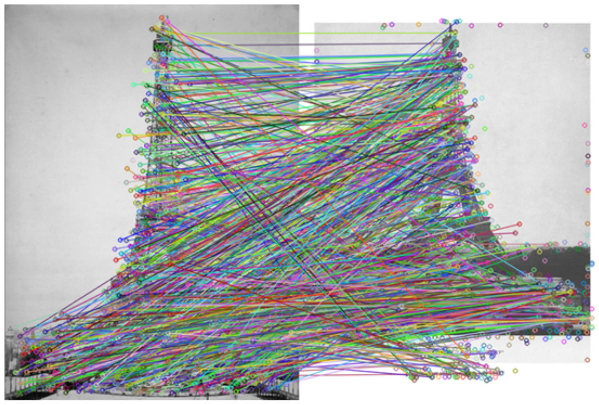


Figure 5: *The SURF keypoints from the image on the left naively matched with those on the right*

to determine a success or failure test result. However, further reductions were required to remove any remaining outliers. The final stage of keypoint matching was to execute a disparity gradient filter. This filter would remove any matches that are inconsistent with the others [Whitehead 2013]. Figure 7 shows that the outliers from Figure 6 were removed.

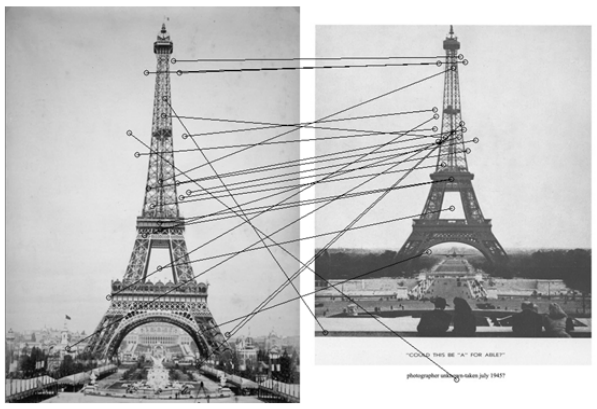


Figure 6: *Matching results using FLANN on two images with SURF keypoints and descriptors*

A test would be considered a success if a significant number of the remaining keypoints were correctly matched. Conversely, a test would be considered a failure if a significant number of the keypoints were incorrectly matched. Figure 7 demonstrates a successful test result, while Figure 8 demonstrates a test that failed. These results were stored as a comparison matrix in a comma separated file. This was to make it simple to import into Excel. All the results were compiled and imported into an Excel worksheet, and formatted to make analysis easier. The pseudocode of this testing process is displayed in Figure 9.

7 Analysis of Feature Matching with Modern to Historial Photos

The following are two tables summarizing the feature matcher accuracies and hamming distances in each landmark, respectively. The hamming distance between a row in one table and a row in another table is the number of times the two rows differ in value. For the hamming distances table, its final row is the sum of all the

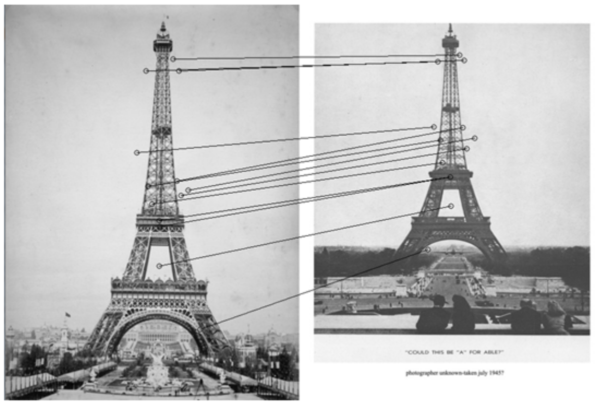

"COULD THIS BE "A" FOR ABLE?"
photographer unknown-taken july 1945?

Figure 7: *Remaining matches after executing a disparity gradient filter*

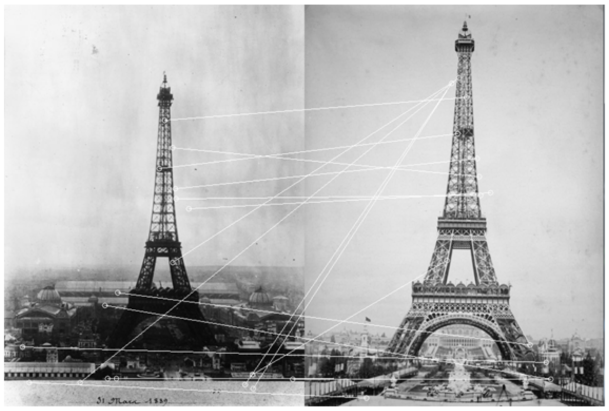
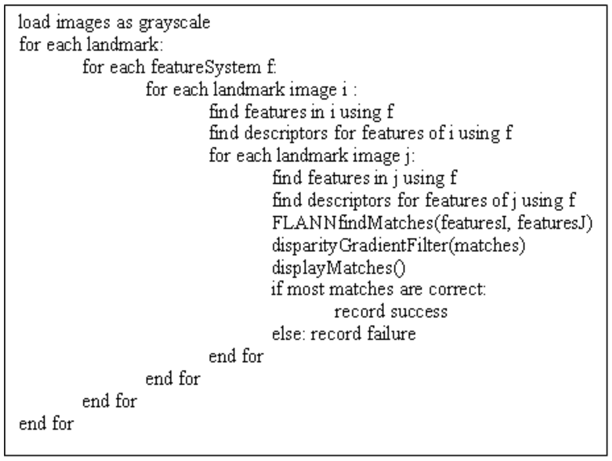


Figure 8: *A test on the Eiffel tower with SURF that failed*

hamming distances for each table comparison for all the landmarks. This provides an overall result of how the tables compare [Whitehead 2013].

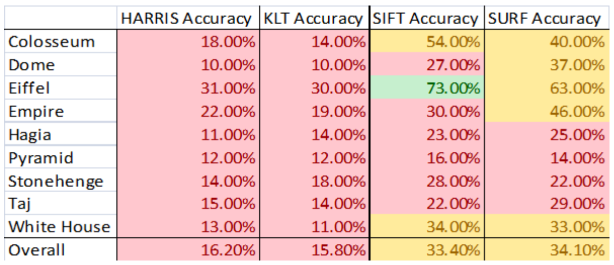
Figure 10 displays a summary of feature matcher accuracies across all the landmarks. Additionally it displays the overall accuracy for each feature matcher, calculated by the total number of successes divided by the total number of tests. Less than 33% will display as red, between 33% and 67% as yellow and above 67% as green. The results demonstrate that SURF was overall the most accurate feature matcher, with SIFT performing only slightly worse. KLT was overall the most inaccurate feature matcher. Figure 11 displays a summary of all the hamming distances. The hamming distance between two feature matchers is the number of times the two techniques had different results. The summary table has a row for each landmark, where the final two rows provide a total for each table comparison as well as the average. These hamming distances are an informative way of comparing the different feature matchers [Whitehead 2013]. The results show that HARRIS and KLT were the most alike in all the landmark results, while HARRIS and SIFT were the least alike of all the landmark results.

The results compiled from all the tables indicate that SIFT and SURF overall outperformed HARRIS and KLT. Additionally, for both SIFT and SURF, the majority of their success results were in the bottom right hand corner of their respective tables. These successful matches correspond to comparisons of modern photographs. This signifies that SIFT and SURF are better at matching when modern images are compared with each other. However, they are



```
load images as grayscale
for each landmark:
    for each featureSystem f:
        for each landmark image i :
            find features in i using f
            find descriptors for features of i using f
            for each landmark image j:
                find features in j using f
                find descriptors for features of j using f
                FLANNfindMatches(featuresI, featuresJ)
                disparityGradientFilter(matches)
                displayMatches()
                if most matches are correct:
                    record success
                else: record failure
            end for
        end for
    end for
end for
```

Figure 9: *Pseudocode of test script used to gather results*



	HARRIS Accuracy	KLT Accuracy	SIFT Accuracy	SURF Accuracy
Colosseum	18.00%	14.00%	54.00%	40.00%
Dome	10.00%	10.00%	27.00%	37.00%
Eiffel	31.00%	30.00%	73.00%	63.00%
Empire	22.00%	19.00%	30.00%	46.00%
Hagia	11.00%	14.00%	23.00%	25.00%
Pyramid	12.00%	12.00%	16.00%	14.00%
Stonehenge	14.00%	18.00%	28.00%	22.00%
Taj	15.00%	14.00%	22.00%	29.00%
White House	13.00%	11.00%	34.00%	33.00%
Overall	16.20%	15.80%	33.40%	34.10%

Figure 10: *Summary of the feature matcher accuracies*

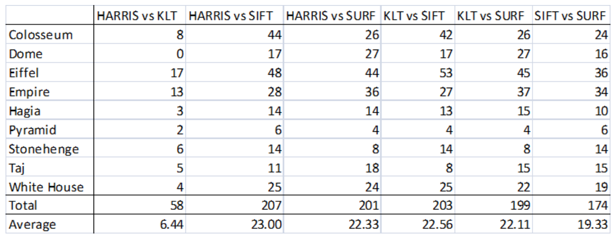
not any better at matching historical photos than HARRIS and KLT, which performed poorly.

A noticeable trend among all the feature systems is that their success results tend to be closer towards the diagonal of their respective tables. These results correspond to image comparisons where the images do not have a significantly large time span. This means that very old images matched with more recent images failed more frequently than images that were only a few decades apart. This could be due to changes in image technology, as well as the transition from film to digital imagery.

8 A proposed improved approach

The proposed technique for improved matching is based off of the Maximally Stable Extremal Regions (MSER) detector. Very briefly, every extremal region is a connected component of the input image thresholded to a specific value [Matas et al. 2002]. These regions represent local minima or maxima of the input image. The set of all MSERs are the extremal regions for each possible threshold value. However, only regions that are maximally stable are kept.

MSER was chosen as a new technique because an abstract feature all the landmarks have in common with their respective photos is their recognizable regions. A feature point from a corner or blob is subject to the technology used to take the photo, which poses a problem for historical photos. In contrast, the way MSER detects regions seemed ideal for this problem. If similar regions could be detected between photos, it would prove to be more accurate than using the techniques previously tested.



	HARRIS vs KLT	HARRIS vs SIFT	HARRIS vs SURF	KLT vs SIFT	KLT vs SURF	SIFT vs SURF
Colosseum	8	44	26	42	26	24
Dome	0	17	27	17	27	16
Eiffel	17	48	44	53	45	36
Empire	13	28	36	27	37	34
Hagia	3	14	14	13	15	10
Pyramid	2	6	4	4	4	6
Stonehenge	6	14	8	14	8	14
Taj	5	11	18	8	15	15
White House	4	25	24	25	22	19
Total	58	207	201	203	199	174
Average	6.44	23.00	22.33	22.56	22.11	19.33

Figure 11: *Summary of the feature matcher hamming distances*

9 Evaluation of new approach

Initially, simple MSER comparisons were executed to determine if accurate matching would be likely. The MSERs were generated for each image, and images were visually compared with each other. The goal was to check if regions would be reasonably similar, such that automatic matching of regions could be done. Preliminary results were mixed. As shown in Figure 12, some landmarks had many similar regions. This could be explained because the landmarks themselves had many repeating shapes, such as windows. Some landmarks, like the one in Figure 13, had relatively few similar regions. This small number would be unhelpful for matching images accurately. As well, some landmarks had virtually no discernibly similar regions. This can be observed in Figure 14.

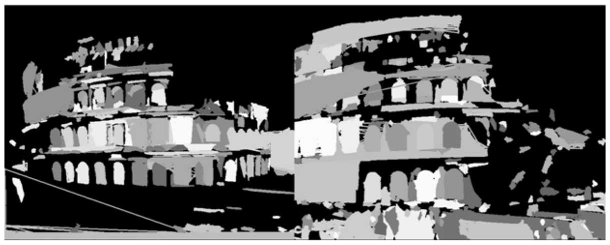


Figure 12: *MSER comparison of the Colosseum with a high number of similar regions*

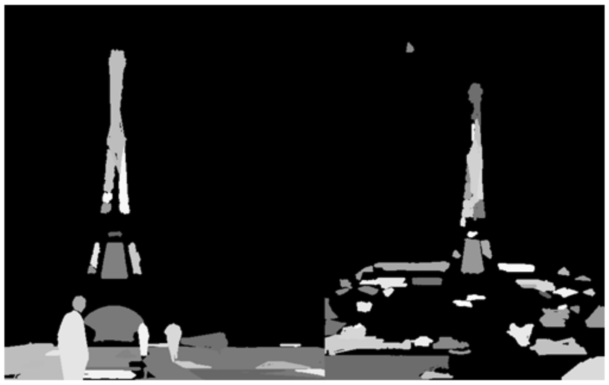


Figure 13: *MSER comparison of Eiffel tower with a low number of similar regions*

It was surprising that there were not numerous similar regions across all the landmarks. Each one had unique areas that could have translated well to comparable regions. For example, Figure 15 shows that the two images that were compared in Figure 14 had a very similar mountain regions.

It was determined that the low number of similar regions could be due to noise in the images. Applying a large Gaussian blur to the images prior to generating MSERs was the next attempt. A Gaussian kernel of size 7x7 with sigma value 0.5 was initially used. On the two images from Figure 15, blurring did not seem to improve the results. Gradually larger values of sigma values were used, with no success. Examples of the results from different sigma values can be seen in Figure 16.

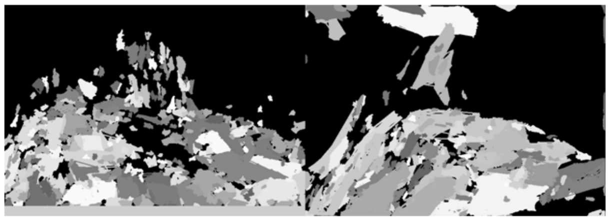


Figure 14: *MSER comparison of Machu Pichu with no discernibly similar regions*

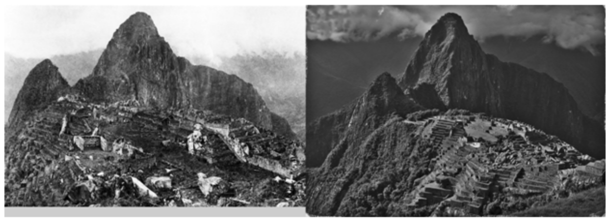


Figure 15: *Original images of Machu Pichu*

In this example, as the sigma value increased, the smaller MSERs disappeared. However, the number of similar regions did not change. This was true for most of the other landmarks. In contrast, blurring landmarks which initially had many similar regions simply sharpened the already existing similar regions. An example of this is demonstrated in 17.

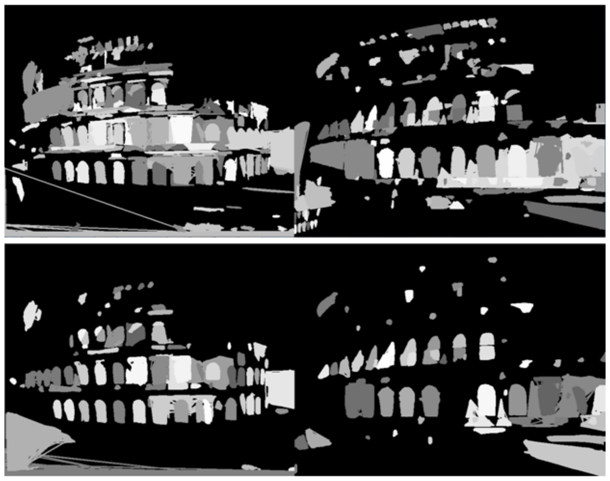


Figure 17: *MSER outputs without prior blurring (above) and with prior blurring (below)*

Changing the kernel size of the Gaussian blur did not seem to affect the outputted MSERs. In the end, it was concluded that some other preprocessing techniques should be applied prior to computing the MSERs. Blurring alone is not enough to consistently isolate similar regions between two images .

10 Summary and Conclusions

The results of the tests confirmed our original hypothesis. Four major feature matchers, HARRIS, KLT, SIFT and SURF all performed poorly when matching historical photos to modern photos. SIFT and SURF were overall the most accurate. However, their successful results can be attributed only to the matches involving modern photos with other modern photos. As well, the feature matchers were more successful only when matching photos with a small time span.

An alternative approach to feature matching was proposed. The expectation was that this approach would solve the problems associated with matching historical photos. Using MSERs instead of corners or blobs initially seemed promising, but no concrete solution was found. The hope is that the suggestion of using MSERs with blurring can be used as a springboard for a larger, complete feature matching solution. This solution would be able to accurately match modern photos with each other, as well as with historical photos.

The overall contributions this project has made to the Computer Vision field are as follows: A useful data set of images has been collected and processed for feature matcher testing. The images for each landmark contain a large range of dates, which can be used to verify the accuracy of any future feature matching solutions. A robust test script has been developed to easily compare the data set and record test results for analysis. Finally, an initial attempt at a more accurate feature matcher has been proposed and described, with hopes that it will be developed further by a future party.

References

BAY, H., ESS, A., TUYTELAARS, T., AND VAN GOOL, L. 2008. Surf: Speeded up robust features. *Computer Vision and Image Understanding 110*, 3, 346–359.

BOUGUET, J. 2000. Pyramidal implementation of the lucas kanade feature tracker. *Intel Corporation, Microprocessor Research Labs*.

HARRIS, C., AND STEPHENS, M. 1988. A combined corner and edge detector. *Proceedings of the 4th Alvey Vision Conference*, 147–151.

LOWE, D. G. 2004. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision 60*, 91–110.

MATAS, J., CHUM, O., URBAN, M., AND PAJDLA, T. 2002. Robust wide baseline stereo from maximally stable extremal regions. *Proceedings of the British Machine Conference, BMVA Press*, 36.1–36.10.

MIKOLAJCZYK, K., AND TUYTELAARS, T. 2008. Local invariant feature detectors: a survey. *Foundations and Trends in Computer Graphics and Vision 3*, 3, 177–280.

MUJA, M., AND LOWE, D. G. 2009. Fast approximate nearest neighbors with automatic algorithm configuration. *VISAPP International Conference on Computer Vision Theory and Applications*, 331–340.

ROTH, G., 2012. Corner (interest point) detection.

SHI, T., AND TOMASI, C. 1994. Good features to track. *Computer Vision and Pattern Recognition*, 593 – 600.

WHITEHEAD, A., 2013. Private communication.

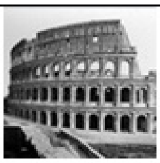
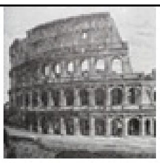
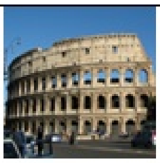
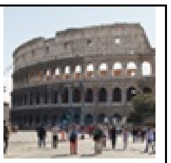
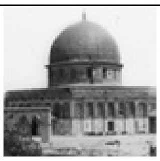
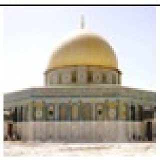
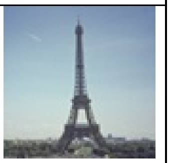
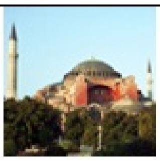
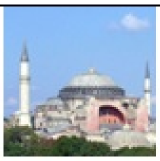
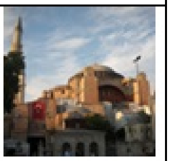
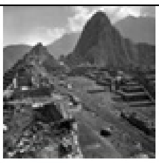
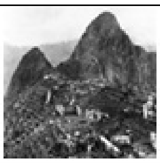
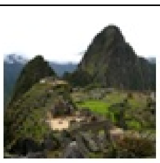
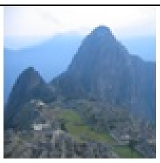
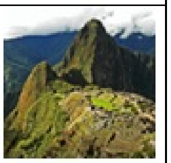
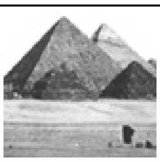
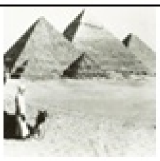
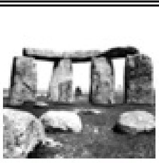
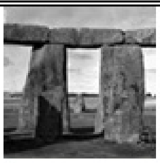
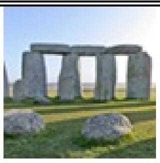
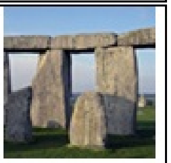
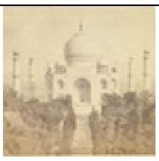
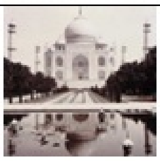
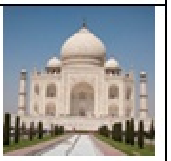
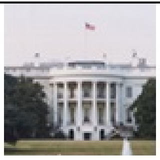
The Colosseum in Rome 1855 - 2011					
Dome of the Rock 1900 - 2012					
The Eiffel Tower 1889 - 2011					
The Empire State Building 1936 - 2011					
Hagia Sophia in Turkey 1890 - 2011					
Machu Pichu 1900 - 2011					
The Egyptian Pyramids 1858 - 2011					
Stonehenge 1887 - 2011					
The Taj Mahal 1874 - 2011					
The White House 1863 - 2012					

Figure 3: *Summary of landmarks used for testing*

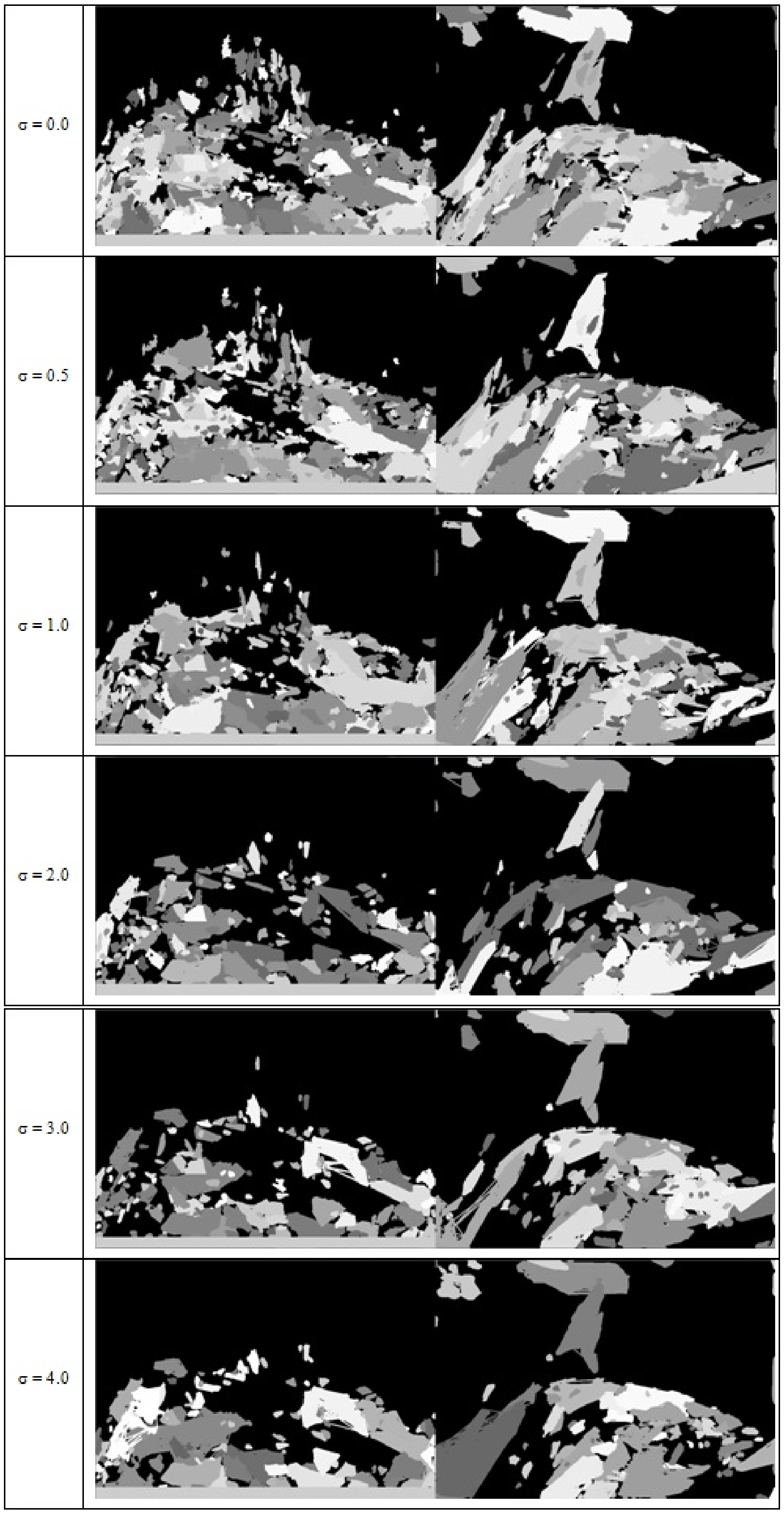


Figure 16: *The effect of blurring on MSERs with different values for sigma*