Scale Invariant Feature Transform: A Graphical Parameter Analysis

Michael May[1]
michael.may@postgrad.manchester.ac.uk
Martin J. Turner[2]
martin.turner@manchester.ac.uk
Tim Morris[1]
tim.morris@manchester.ac.uk

[1] The School of Computer Science
The University of Manchester
Manchester, UK

[2] Research Computing Services
The University of Manchester
Manchester, UK

Abstract

In this paper we introduce a general purpose graphical processing unit (GPGPU) based method for performing a sweep across a set of the scale invariant feature transform (SIFT) parameters for pairs of images. The focus of the paper is the analysis of the data generated using information visualisation techniques including a cross brushing technique between parallel coordinates, scatter plots and histograms. Results have shown us the importance of carefully selecting some parameters depending upon the properties of an image pair while other parameters are shown to be robust to variation. The parameters chosen by analysis of the sweep data have then been compared to the previously published SIFT's values and a consistent improvement in accuracy is shown.

1 Introduction

The scale invariant feature transform (SIFT) is a feature detection algorithm used for finding correspondence between parts of images thereby allowing image matching. The algorithm generates high dimensional features from patches selected based on pixel values which can then be compared and matched to other features. The algorithm has a set of parameters which can be varied to alter how it behaves and the choice and modification of current favoured values can be used to improve the quality of the results. In the original paper by David Lowe [1] a set of default parameters is given with a variety of images but whether or not these are optimal is not clear.

This paper shows the results of sweeps across this parameter space for various images in an effort to find the best parameter selection for differing scenarios. A semi-exhaustive search has been completed by utilising the speed-up provided by a cluster of general purpose graphical processing units (GPGPUs) over CPUs. The large amount of data produced has then been analysed using parallel coordinate graphs [8], scatter plots and histograms to uncover patterns to indicate how individual parameters effect the algorithm's accuracy.

2 Scale Invariant Feature Transform

The original SIFT feature detection algorithm developed and pioneered by David Lowe [] is a four stage process that creates unique and highly descriptive features from an image. These features are designed to be invariant to rotation and are robust to changes in scale, illumination, noise and small changes in viewpoint.

The features can be used to indicate if there is any correspondence between areas within images. Clusters of features from an image that are similar to a cluster of features from another image may indicate, with a high likelihood, areas that match. This allows object recognition to be implemented by comparing features generated from input images to features generated from images of target objects. The four stages of the SIFT algorithm are as follows, full details of which are given in Lowe's paper []:

1. **Scale-space extrema detection.** The first step is to create the Gaussian scale-space pyramid. Successive blurred images are produced from the convolution of Gaussian functions to create multiple octaves. The difference of Gaussian (DoG) is calculated as the difference between two consecutive images within an octave. The initial set of candidate features are selected by comparing each point in the DoG images to its 26 neighbours and looking for extrema.

2. **Feature localisation.** The number of features is reduced in this stage by reducing the number of features. Interpolation occurs to locate the exact, sub-pixel, location of the candidate features before eliminating the points that are in areas of low contrast and those that are localised along edges.

3. **Orientation assignment.** One or more orientations for each feature is calculated, a process which results in the rotational invariance of the descriptor. The image gradient directions of the pixels in a feature's neighbourhood are calculated and added to an orientation histogram with 36 bins. The values in the neighbourhood are Gaussian weighted so those nearer the centre have a greater effect on the resulting orientation.

4. **Creating the feature descriptor.** The feature descriptor is a 128 dimensional vector which describes the pixel properties of the area surrounding a feature. A 4×4 array of 16 histograms is centred on the feature and rotated to match the orientation calculated in the previous step. The gradient magnitudes are given a Gaussian weighting, added to the histograms and normalised to create the descriptor.

To match features often the Euclidean distance between two feature vectors is used to find the nearest neighbour.

3 SIFT Parameters

The choice of parameter values of SIFT effect the response of the algorithm but exactly how changes in their values vary the result and accuracy of feature matching has not previously been studied in sufficient detail. Table 1 shows a list of the main intrinsic parameters which control the response of the algorithm and Lowe's default parameters []. A subset of these have been selected as the focus of the parameter sweep. The parameter sweep is the incremental adjustment of the parameter values with the output of the algorithm recorded for each change.

Parameter	Description	Default Value
Octaves (1)	The number of octaves.	3
Intervals (1)	The number of sampled intervals per octave.	3
Sigma (1)	The sigma value for initial Gaussian smoothing.	1.5
Image doubled (1)	Whether to double the image size before pyramid construction?	Yes
Initial sigma (1)	The assumed Gaussian blur for input image.	0.5
Contrast threshold (2)	The threshold on feature contrast $\|D(x)\|$ (minimum).	7.7 (0.03)[1]
Curvature threshold (2)	The threshold on feature ratio of principle curvatures (maximum).	10
Orientation histogram bins (3)	The number of bins in histogram for orientation assignment.	36
Orientation sigma factor (3)	This determines the Gaussian sigma for orientation assignment.	1.5
Orientation radius (3)	This determines the radius of the region used in orientation assignment.	3.0 × Ori_Sig_Fctr
Orientation peak ratio (3)	The magnitude relative to maximum resulting in multiple orientations.	0.8
Descriptor histogram width (4)	The height and width of the descriptor histogram array.	4
Descriptor histogram bins (4)	The number of orientation bins per histogram in descriptor array.	8
Descriptor width (4)	The height and width of the descriptor.	16
Descriptor magnitude threshold (4)	The threshold on the magnitude of the elements of the descriptor vector.	0.2
Feature vector (4)	The dimensions of the feature vector	128
Match ratio	The ratio of the nearest to next nearest feature during matching.	0.8

Table 1: The main parameters of the SIFT algorithm and Lowe's default values. The number in brackets refers to the stage of the SIFT algorithm where the parameter is applied.

Often experiments use the original Lowe algorithm parameters without specifically tuning them for the task [1, 3, 12, 15] and these may not provide the best results. It has not been shown that the Lowe parameters are the best generic parameters even though they are a set which appear to work satisfactorily for many cases.

Other papers have varied the parameters for their work. Jagadish and Sinzinger [9] selected a match ratio of 0.6 for their work comparing SIFT to Radial Feature Descriptors on tone mapped images without explanation as to why this value was selected. This is also the case in the paper by Battiato et al. [2] who justify the change of the match ratio through experimentation. They also find that adjusting the contrast threshold to extract fewer points results in a smaller set of more stable features. The paper by Park et al. [16] uses SIFT for fingerprint identification and chooses to use 4 octaves with 5 intervals and a Gaussian sigma of 1.8. A paper by Tang et al. [17] shows that increasing the Gaussian smoothing reduces the number of features generated from an image. A paper by Cesetti et al. [5] automatically adjusts the contrast threshold value based on the properties of the images. An equation calculates a contrast threshold based on the intensity and size of the image and the image is not processed at scales where this value becomes too small as it proposes that there is a low probability of finding useful features in a low contrast image. Other papers focus on techniques for tuning parameters for feature detectors and descriptors including SIFT, DAISY [10] and GLOH [14] using various methods [7, 13, 18, 19].

These cases indicate that adjustment of the parameters can be beneficial to the results and that Lowe's defaults are not always optimal. However, they do not provide a full overview of how to intelligently choose the best parameters for a scenario nor do they cover all the available parameters.

4 GPGPUs and CUDA

The parameter sweep is a computationally expensive task as adjusting each parameter through a range of values means the SIFT algorithm will have to be executed on a pair of images for each iteration to see the effects of all the possible parameter states. This is too time consum-

[1] The contrast ratio value depends on the image representation; [0, 256] or [0, 1]. The two values are equivalent. For our experiment we use [0, 256] hence the values are larger than in Lowe's paper.

ing to be carried out on a CPU in a reasonable time, so general purpose graphics processing units (GPGPUs) have been used to implement SIFT. The inherent parallelism of many parts of the SIFT algorithm means it lends itself to being implemented on GPGPUs resulting in significant speed-ups. Tests on a GPU have shown a speed increase for SIFT of up to twenty times over a CPU[2].

The reason why the GPU is so powerful and can be utilised for this project stems from the large amounts of money being invested in improving their performance for the games industry. They are mass produced and relatively cheap and have the ability to perform highly parallel floating point calculations. NVIDIA's CUDA is a general purpose parallel computing architecture that provides the tools required for the coding of parallel code for a GPU and facilitates its execution in a fraction of the time it would take to execute on a CPU. The code allows homogeneous execution on both the CPU and GPU so all the resources of the system can be taken advantage of and code which is suited to serial execution can still be executed on the CPU. The architecture also allows the use of multiple GPUs in parallel.

SIFT has been shown to be successfully parallelised on the GPU in several cases. These include the use of CUDA in the cases of CudaSIFT [4] and SiftGPU [20], and the use of OpenGL textures to store and process the images [6].

5 Methodology

To perform the parameter sweep a pair of annotated images are required. The areas which match between the images are annotated by hand so the system can tell where the scene should show correspondence. This is shown in figure 1. The system is based on CudaSIFT by Marten Bjorkman [4] and extracts the features from each of the images in parallel on two independent GPUs. The extracted features are then matched on a single GPU and the number of correctly and incorrectly matched features can be calculated using the annotation points. Then a parameter is changed and the process is repeated.

Parameter	Starting value	Samples	Step size	Final value
Sigma	0.1	5	0.6	2.5
Contrast threshold	1	5	5	21
Curvature threshold	5	5	4	21
Intervals	2	5	1	6
Octaves	2	4	1	5
Orientation peak ratio	0.1	5	0.2	0.9
Descriptor magnitude threshold	0.1	5	0.2	0.9
Match ratio	0.2	5	0.2	1.0

Table 2: The sweep input parameters.

A subset of the parameters in table 1 has been used within these experiments. The parameters chosen are shown in table 2 along with their starting values, the range over which they are varied and the step size of each iteration. These sweep values were selected through initial experiment, calculating the computation required, by studying how the algorithm works and recommendations from related papers. This set of parameters results in up to 312500 iterations of the algorithm, depending on the features generated, and takes approximately 20 hours for an image pair [3].

The images used are varied so that different objects are detected in various scenes with changes in scale, rotation and viewpoint so that many different possible SIFT usage scenarios

[2]Using an AMD Athlon64 FX-70 CPU and an NVIDIA GeForce 8800 GTX GPU.

[3]Using an Intel Core i7-920 2.66GHz CPU and an NVIDIA 9800 GX2 GPU.

Parameter	Description
Points 1	The number of points extracted from the first image
Points 2	The number of points extracted from the second image
Total matches	The total number of matches between the images
Correct matches	The number of correct matches between the annotated regions
Annotated matches	The total number of matches from the annotated regions in the first image to the second image
Accuracy	The percentage of correct matches in the annotated matches

Table 3: The sweep output parameters.

are covered. This will help indicate which parameters effect the algorithm differently under different circumstances, provide more information about the optimal parameters and help indicate any trends and correlation across the parameters. The data produced for information visualisation analysis has 14 dimensions; one for each of the input and output parameters. There are six output parameters that are generated during the parameter sweep and these are described in table 3.

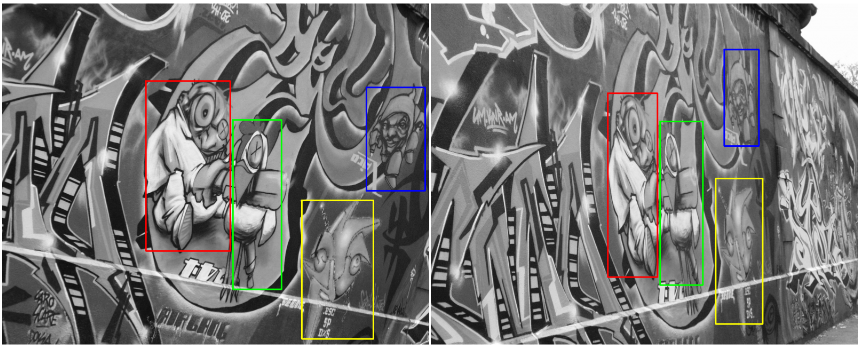

Figure 1: An example of an image pair with annotation boxes showing corresponding regions.

6 Process Employing Information Visualization Techniques to Gain Insight

This section outlines how the interactive analysis process for analysing results of the parameter sweeps[4]. Parallel coordinate graphs [8], scatter plot arrays and histograms have been used to visualise and analyse the data. Parallel coordinates is a common way of visualising multivariate data such as that produced by the parameter sweep. Each parameter has its own parallel axis and a polygonal line with vertices on the parallel axes represent a point in n-dimensional space. This visualisation method allows correlation between parameters to be viewed with the careful use of brushing.

An example of a parallel coordinate graph is shown in figure 2. It has been brushed to display parameter combinations with the highest accuracy and a number of correct matches greater than 10. Brushing is an interactive process of reducing the data to a subset. It is done by selecting parameters value ranges and data values outside of these ranges are excluded. The red lines display the parameters which meet this criteria. A parameter must produce a high percentage accuracy and a number of correct matches greater than a minimum to be

[4]The full set of images and output data files are available at http://www.rcs.manchester.ac.uk/aboutus/students/may.

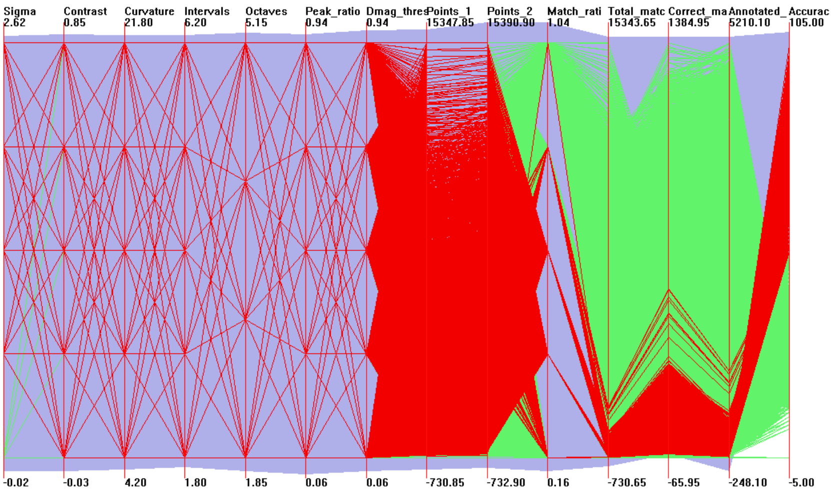

Figure 2: Parallel coordinates of a 14 dimensional dataset with 262500 elements. The data is brushed to exclude elements with a very low number of correct matches and low accuracy.

deemed a good selection. The reason that a minimum number of correct matches is required is that a single correct match could give an accuracy of 100 percent but would be useless for confirming matches between images as a cluster of points are required and a single data point could be erroneous. Setting this minimum at 10 eliminates parameter combinations which give a high accuracy without enough data to be confident of an image match.

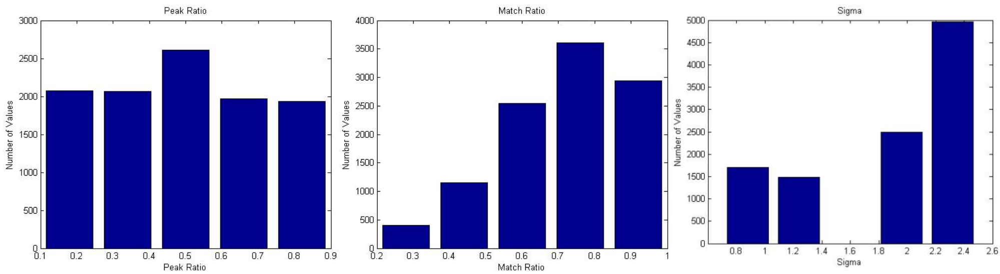

Figure 3: Histograms of three parameters after the data has been brushed. The peaks indicate the values which contribute most to the data selected by brushing.

On the parallel coordinate graph multiple overlaid lines on a parameter point cannot be distinguished from a single line through a point. The use of histograms allows each parameter to be plotted individually showing the distribution of values that pass through each parameter point. This indicates which parameters contribute most to the results with the highest accuracy as shown in figure 3. The scatter plot array, such as that in figure 4, shows each parameter plotted on a 2D graph against every other. This allows correlation between individual parameters to be observed.

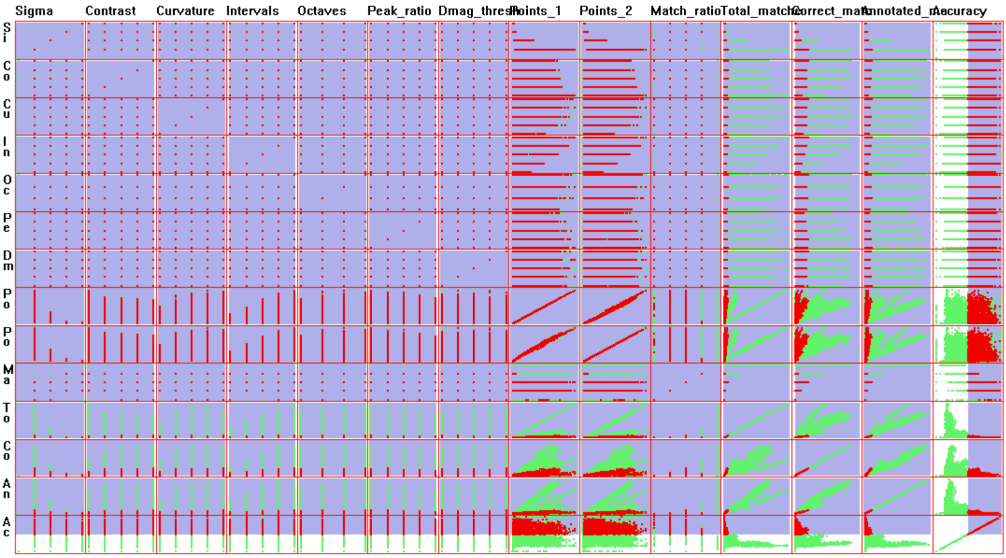

Figure 4: Scatterplot array of the dataset shown in figure 2 with the same brushing applied.

7 Analysis and Recommendations

This section outlines and explains the results of parameter sweeps using 10 image pairs which range in size from 300 kilopixels to 1.2 megapixels. The data has been brushed to display at least 500 parameters combinations with the highest accuracy. This is done by adjusting an accuracy threshold until the resulting number parameters combinations are greater than 500. In conjunction with this the number of correct matches is brushed to be greater than 10 as explained in the previous section. The remaining parameters give an indication of how to parametrise SIFT in order to achieve reasonably high accuracy matches. Table 5 shows the maximum histogram values of the parameters after brushing, i.e. the parameter values which most often result in a high accuracy. The data is explained in detail in relation to each parameter below.

	ball	**book**	**car**	**landscape**	**left-right**	**lowe**	**soup bin**	**soup b**	**stick**	**wall**	**mean**
Correct Min	10	10	10	10	10	10	5	5	5	10	
Accuracy Min	40	20	10	30	50	50	8	16	7	19	
Sigma	0.7	2.5	2.5	1.9	0.7	0.7	1.9	2.5	2.5	0.7	1.7
Contrast	6	16	6	6	6	21	6	6	6	11	9.0
Curvature	13	5	9	21	21	5	5	5	17	9	11.0
Octaves	3	2	3	4	2	3	2	2	4	4	2.9
Intervals	2	2	3	4	4	2	2	2	5	2	2.8
Peak Ratio	0.1	0.7	0.3	0.1	0.1	0.7	0.3	0.1	0.5	0.1	0.3
Dmag Thresh	0.1	0.1	0.7	0.7	0.3	0.1	0.1	0.1	0.1	0.7	0.3
Match Ratio	0.6	1	0.8	1	0.6	0.8	1	0.8	1	0.8	0.8

Figure 5: Maximum histogram values after brushing the minimum accuracy and correct matches values. The image pair IDs are in bold. The colour indicates if the peak is strong (a single very clear peak), medium (clearly the largest peak but with other large peaks present) or if the histogram is relatively flat as illustrated in figure 3.

Sigma. Sigma has strong peaks across the full range of values meaning that the choice of sigma is very specific to an image type and can greatly effect the results. Sigma therefore needs to be chosen carefully and a single value cannot guarantee accuracy across all image types. A very low sigma, for example 0.1, will not generate features as the images in the difference of Gaussian stage will not be different enough for there to be edges to be generated by their subtraction. Once a minimum sigma value has been reached features will be created and then as sigma is increased further the number of features will gradually be reduced as image blur increases.

Contrast threshold. Contrast threshold, like sigma, consists of strong peaks across the full range of values and therefore must also be chosen carefully for each image pair that is used. A bad choice can cause low accuracy and the best value varies from image to image so a single, universal, value will not suffice. A value of 6 appears regularly across the image pairs as the best choice.

Curvature threshold. The maximum curvature threshold ranges between 5 and 21 and is not consistent across the various histograms. It appears that that its choice can vary in importance, with most histograms having medium peaks. The mean is 11, which is very close to 10 the value proposed by Lowe. The number of features remaining increases as the parameter is increased so setting it low will generate fewer features.

Octaves and intervals. The optimal number of octaves and intervals is 4 or less in nearly all cases. A higher number of octaves and intervals appears to be unnecessary as they include the features generated when the parameter is set to a lower value. The extra features generated from the increase in octaves or intervals will be created when the images are smoothed and scaled more and therefore comparatively few extra features will be extracted. The objects matched in the test images vary in scale substantially and parameters values of 4 of less suffice. Most of the histograms are quite flat and and so selecting the non-optimal value may not be detrimental to the accuracy.

Orientation peak ratio. This generates very flat histograms therefore the value chosen does not appear to effect the algorithm. This indicates that the extra computational process and costs of creating a secondary peak is effectively unimportant, as the number of features where this is done is too insignificant to effect the results or that as long as a secondary feature is created in some cases where the peak ratio is greater than 0.9 then generating features for peaks with lower ratios does not make a significant difference. Further tests will compare the results of the algorithm with this stage removed completely to see how the algorithm is effected.

Descriptor magnitude threshold. The results consist of medium and flat histograms. All the cases where the histograms have some stronger peaks are when the optimal parameter value is low, 0.1 to 0.3. When high values appear in the results the histograms are very flat and the choice of parameter makes little difference. This means that selection of a lower value, for example Lowe's defaults of 0.2, will allow for high accuracy for all the images in this set.

Match ratio. Strong and medium histogram peaks indicate that the match ratio should be high, 0.8 on average. The lower the match ratio value is the more discriminative it is reducing the number of matches. Extremely low numbers of correct matches have been brushed out of the data as they do not provide sufficient information to reliably indicate correspondence between images. This explains why the match ratio value tends to be high and why it is better to select a larger value.

When the parameter selection results in a high number of features being extracted this is generally detrimental to accuracy. High numbers of features are more likely to result in mismatches as there are more opportunities for the features to match incorrectly. Even if it does not effect the accuracy it is beneficial to avoid unnecessary feature extraction as this results in more computation. A balance must be found between generating enough features to match the target area within the image and too many such that mismatches become more likely to occur and become overwhelming computationally. The use of parallel coordinates, scatterplots and histograms which are brushed to remove low accuracy points then show that many of the parameters combinations which result in a high number of extracted points are automatically discarded.

A point to note is in relation to the images labeled **soup-bin**, **soup-b** and **wall** and the data relating to these. For these the minimum correct matches was reduced to a lower value, 5, as the accuracy when this was set to 10 was too low and there was not enough data left after brushing to draw a conclusion as to the best parameters. This is due to the objects matched in the images being smaller than in the others and as such a lower number of correct matches were obtained from the images. 5 is still a sufficient number to identify the object correctly. Also it is interesting that **soup-bin** and **soup-b**, which both use the same image for matching to different scenes have very similar parameter sweep results.

It should also be noted that while these sweeps took up to 24 hours to complete the time can be reduced. By removing two parameters from the sweep and setting their values to Lowe's defaults the execution time can be reduced to half an hour. The two parameters that can be safely discounted are the orientation peak ratio and descriptor magnitude threshold as varying their values doesn't appear to effect the accuracy greatly. This can be seen in table 5 which shows that the histograms produced for these parameters are generally quite flat with few significant peaks.

7.1 Parameter Testing

Table 4 shows the percentage accuracy of the algorithm on the test image pairs when using Lowe's default values and the values generated from the parameter sweeps. It shows that in most cases the parameter values obtained from the sweep and graphical analysis perform equally well or better and some quite significantly so. The cases with low accuracy results indicates that the image pair does not respond well to the SIFT in general due to the properties of the image pair and these are generally the cases where the parameter sweep does not improve the results.

Parameter	**ball**	**book**	**car**	**landscape**	**left-right**	**lowe**	**soup bin**	**soup b**	**stick**	**wall**
Lowe	7	4	8	8	52	4	5	3	2	8
Sweep	49	0	46	44	51	76	2	0	10	37

Table 4: The percentage accuracy for feature matching when using the Lowe's default SIFT parameters and the parameters obtained in the sweep. The sweep parameter values used vary for each image pair as shown in table 5.

8 Conclusions

This interactive technique, it is proposed, should be used to fine tune SIFT in situations where the images are of a known type rather than the results in this paper being a solution to choosing the best parameters for all occasions. The data in table 4 supports this and highlights the success of the technique. The results show that some parameters such as sigma and the contrast threshold have strong peaks over the range of values meaning that a single selection will never suit all situations. It is therefore important to choose such parameters well to ensure high accuracy. Many of the other parameters are quite flexible and robust which means that non-optimum selection may not be detrimental.

Further work will look into how to use this technique to create an intelligent means of parametrising SIFT based on the properties of an image pair. The aim of this is to allow a SIFT user to reliably set parameters based on the image properties such as the size of the object, viewpoint or the object type without having to apply this parameter sweep technique themselves. Other areas of interest include the effect of other parameters such as the number of bins in the descriptor, test other ranges and step values and other images types such as high dynamic range (HDR) and infrared.

Overall, the parameters of SIFT cannot make an improvement by adjustment if the data is not within the image in the first place thus there is an inherent best match accuracy based on the image data. However, selecting the wrong parameters can reduce the accuracy of SIFT as shown in 4. Tuning has been shown to make improvements which may be beneficial to an application with constrained bounds and where the task will be repeated many times to justify the computationally expensive sweep process. It is also proposed that it is beneficial to use this technique with new untested image types and scenes to generate initial parameter estimates.

References

[1] A E Abdel-Hakim and A A Farag. CSIFT: A SIFT Descriptor with Color Invariant Characteristics. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume Volume 2, pages 1978–1983, 2006. ISBN ISSN:1063-6919.

[2] S Battiato, G Gallo, G Puglisi, and S Scellato. SIFT Features Tracking for Video Stabilization. *14th International Conference on Image Analysis and Processing*, pages 825–830, September 2007.

[3] M Bicego, A Lagorio, E Grosso, and M Tistarelli. *On the Use of SIFT Features for Face Authentication*, volume 00. IEEE, 2006.

[4] M Bjorkman. Computer Vision and Active Perception Lab, 2008. URL http://www.csc.kth.se/~celle/.

[5] A Cesetti, E Frontoni, A Mancini, P Zingaretti, and S Longhi. A Vision-Based Guidance System for UAV Navigation and Safe Landing using Natural Landmarks. *Journal of Intelligent and Robotic Systems*, 57(1-4):233–257, 2009. ISSN 0921-0296.

[6] S Heymann, K Maller, A Smolic, B Froehlich, and T Wiegand. SIFT implementation and optimization for general-purpose GPU. In *Proceedings of the International Con-*

ference in Central Europe on Computer Graphics, Visualization and Computer Vision, 2007.

[7] G Hua, M Brown, and S Winder. Discriminant embedding for local image descriptors. In *IEEE 11th International Conference on Computer Vision*, volume Internatio, pages 1–8. Citeseer, 2007.

[8] A Inselberg. *Parallel Coordinates - Visual Multidimensional Geometry and Its Applications*. Springer, 2009.

[9] K Jagadish and E Sinzinger. Image Matching Using High Dynamic Range Images and Radial Feature Descriptors. In *Advances in Visual Computing*, pages 359–369. 2008.

[10] V Lepetit. A Fast Local Descriptor for Dense Matching, 2008.

[11] D G Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.

[12] J Luo, Y Ma, E Takikawa, S Lao, M Kawade, and B Lu. Person-specific sift features for face recognition. *Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop*, pages 4–7, 2007.

[13] K Mikolajczyk and J Matas. Improving descriptors for fast tree matching by optimal linear projection. In *IEEE 11th International Conference on Computer Vision*, volume IEEE 11th, pages 1–8, 2007.

[14] K Mikolajczyk and C Schmid. A Performance Evaluation of Local Descriptors. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1615–1630, 2005.

[15] K Mikolajczyk, T Tuytelaars, C Schmid, A Zisserman, J Matas, F Schaffalitzky, T Kadir, and L V Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65:43–72, 2005.

[16] U Park, S Pankanti, and A K Jain. Fingerprint verification using SIFT features. *Proceedings of SPIE*, 2008. ISSN 0277786X.

[17] C Y Tang, Y L Wu, M K Hor, and W H Wang. Modified sift descriptor for image matching under interference. In *Machine Learning and Cybernetics, 2008 International Conference on*, volume 6, 2008.

[18] S Winder and M Brown. Learning local image descriptors. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[19] S Winder and M Brown. Picking the best DAISY. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 178–185, 2009.

[20] Changchang Wu. SIFT on GPU (siftgpu), 2007. URL http://www.cs.unc.edu/~ccwu/siftgpu/.