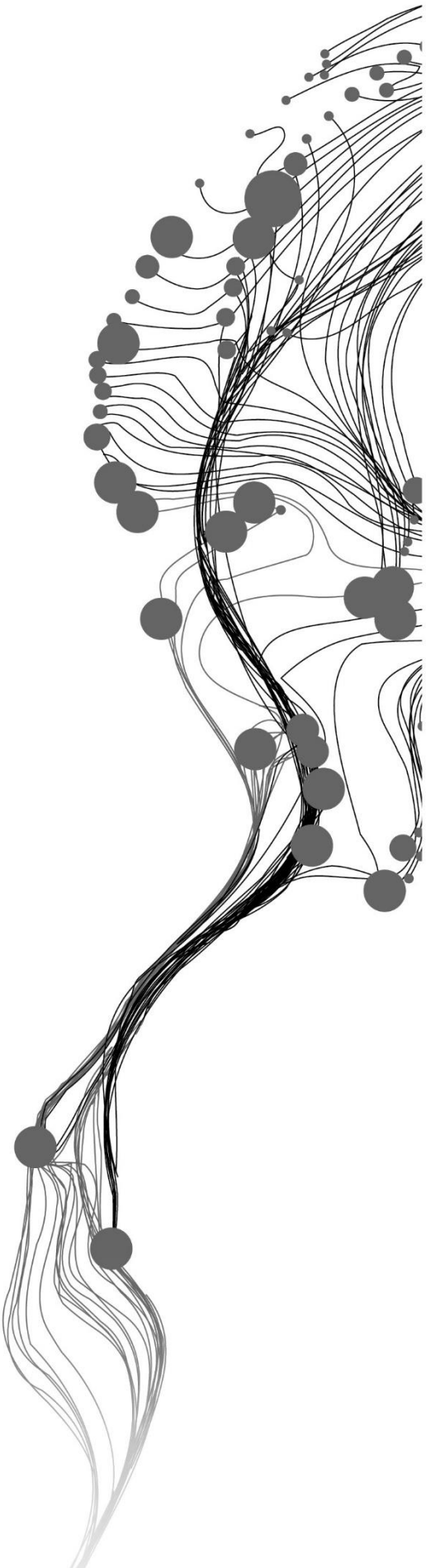
MULTI-RESOLUTION AUTOMATED IMAGE REGISTRATION

Fredrick Arthur Onyango
February, 2017

SUPERVISORS:
Dr. -Ing. Francesco Nex
Dr. -Ing. Michael Peter

ADVISOR:
Mr. Phillipp Jende MSc.



MULTI-RESOLUTION AUTOMATED IMAGE REGISTRATION

Fredrick Arthur Onyango
Enschede, the Netherlands, February, 2017

Thesis submitted to the Faculty of Geo-Information Science and Earth Observation of the University of Twente in partial fulfilment of the requirements for the degree of Master of Science in Geo-information Science and Earth Observation.
Specialization: MSc. Geoinformatics

SUPERVISORS:
Dr. -Ing. Francesco Nex
Dr. -Ing. Michael Peter

ADVISOR:
Mr. Phillipp Jende MSc.

THESIS ASSESSMENT BOARD:
Prof. Dr. Ir. M.G. Vosselman (Chair)
Prof. Dr. –Ing. M. Gerke (External examiner, Institute of Geodesy und Photogrammetry, Technische Universität Braunschweig)

ABSTRACT

The acquisition of images in the field of photogrammetry has developed rapidly over the past decades. The resultant images have varied resolutions due to the different platforms and cameras used to acquire these images. Manned aircrafts have for a long time been used to capture aerial images for photogrammetric applications like topographical mapping, but this mode of image acquisition has proved to be costly. Unmanned Aerial Vehicles (UAV) have now gained popularity due their use in acquiring low cost and high resolution images. Researchers from various fields have utilised the advantages of UAV images to generate high resolution 3D models of captured scenes and this process makes use of image registration techniques used to find correspondences between a pair of overlapping images. Generation of multi-resolution 3D models presents an interesting application that requires multi-resolution images capturing the same scene. This research addresses the problem of registering multi-resolution images, in particular, aerial oblique and UAV images. An investigation is done on the state-of-the-art feature detector/descriptors and feature matching strategies so as to identify a promising methodology that can be used to register UAV images to aerial images. The registration result is a fundamental matrix that represents the geometrical relationship between the image pair that can be used to relatively orient the UAV image with respect to the aerial image. Preliminary tests were conducted using SIFT, SURF, KAZE, SURF/BRIEF, BRISK and AKAZE feature detector/descriptors on a pair of images. Results show that AKAZE outperforms SIFT, SURF, KAZE, SURF/BRIEF and BRISK by producing more matches than the other detectors. AKAZE was then parametrised and an automatic procedure was developed to register the image pair. Part of the procedure involved the computation of multiple homographies between the images so as to identify common planes which led to a reduction in the number of incorrect matches iteratively. The developed procedure was then applied to image pairs taken under different viewing angles and a different scene so as to evaluate its performance. The results demonstrate that the developed methodology yields favourable results and this is evident from the results after evaluating its performance and assessing the accuracy of the F matrix.

Keywords: Multi-resolution, image registration, aerial image, UAV image, feature detection, feature matching, homography, fundamental matrix

ACKNOWLEDGEMENTS

First and foremost, I'd like to extend my sincere gratitude to Dr. -Ing. Francesco Nex, Dr. -Ing. Michael Peter and Phillipp Jende who have been instrumental in offering sound advice, invaluable feedback and constructive criticism throughout the entire period of this research.

Secondly, I'd like to thank all my GFM colleagues who took their time to listen to the challenges I was facing and offering solutions to the problems I was facing.

Special thanks goes to my family, relatives and friends back home in Kenya who were always there for me by keeping in touch and wishing me all the best in my studies. All the Skype calls, phone calls and messages gave me reason to soldier on with my studies.

Utmost gratitude goes to my employer who accepted to grant me a study leave. I'll be forever grateful for this opportunity that will go a long way in shaping my career.

To the new friends – Marjolein, Ken, Patrick, Mutinda, Dan, Nick, Loise, Callisto, Eliza, Jacob, Benson, Grachen, Mariam, Petulo, the list is endless – I made since my arrival, you have been like family to me. I'll forever cherish the social moments we had together and the good times we shared to make the thesis journey bearable.

Finally, I'd like to thank the Netherlands Fellowship Programme for providing me with a scholarship to pursue my life-long dream of studying abroad. I learnt a lot during my stay in the Netherlands and I hope you continue awarding scholarships to more students around the world.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1. Motivation and problem statement

During the last decades, image acquisition devices have developed rapidly and they have acquired a lot of images that have diverse characteristics such as a wide range of resolutions. Manned aircrafts are being used to capture aerial images for aerial surveys. This method has proved to be quite costly but offers images that cover large areas due to the wide field of view of the cameras used and the aircraft's flying height. Unmanned Aerial Vehicles (UAVs) are being used to acquire images for various civil and topographic mapping applications. These systems provide a low-cost alternative to the traditional airplanes as platforms for spatial data acquisition (Nex & Remondino, 2014). They tend to have high repeatability and flexibility in data acquisition making them popular platforms for image acquisition. To add to that, UAVs acquire images that have a Ground Sampling Distance (GSD) of up to 1 cm which is considered relatively high compared to images taken by manned aircrafts. Other image acquisition devices are digital handheld cameras and smartphones which are off the shelf products. They are often used to take terrestrial photos of a scene.

UAVs are now offering promising technologies that are bridging the gap between terrestrial and traditional aerial image acquisitions (Nex et al., 2015). Recent developments of image acquisition devices have led to fast and inexpensive acquisition of high resolution images. Researchers from various disciplines have utilised this advantage to generate 3D models of cultural heritage sites, urban cities, disaster scenes etc., from 2D images. This process is possible when multiple images of a scene are taken from different viewpoints around the scene of interest. When an object has a complex architecture such as intrusions or extrusions, then UAVs can be used to acquire images at favourable viewpoints to minimise occlusions (Gerke, Nex, & Jende, 2016). Where a continuous model of a scene is required at different resolutions, then high resolution terrestrial and UAV images can be integrated with lower resolution airborne oblique images.

Using only one type of image dataset to generate 3D scenes may not deliver seamless products. For instance, when only terrestrial images are used to generate a 3D model of a building then the roof, parts of a balcony and other structures that are only visible from an aerial perspective will not be captured. In case the aerial oblique images are used, then the 3D model will have a low resolution and building parts like the underside of a balcony will be occluded. Similarly, when only oblique UAV images are used, the generated 3D model will have a high resolution but will have occlusions like the underside of balconies and roof gutters.

The integration of these different kinds of images that vary in resolution is interesting but problematic and it is considered unsolved (Gerke et al., 2016). A crucial part in trying to solve this problem involves identifying correspondences between these images. This process is known as image registration. Goshtasby (2012) defined it as *"the process of spatially aligning two images of a scene so that corresponding points assume the same coordinates"*. This process is crucial in the field of photogrammetry because it aides in the identification of tie points which is crucial for retrieving the images' relative orientation.

Finding these correspondences can be done manually but this is time consuming and labour-intensive, hence the need for automation emerged which has led to the development of automatic image registration algorithms. However, there is no universal method for image registration because images may have different characteristics in terms of geometry, radiometry and resolution (Zitová & Flusser, 2003; Shan et al., 2015). Figure 1.1 shows an example of an aerial oblique, UAV and terrestrial image. The figure illustrates the challenges faced. First, airborne oblique images are taken at a different angle and altitude compared to oblique UAV images. This introduces the difference in scale and viewpoints which affects the performance of registration algorithms. Secondly, the lighting conditions are also different, posing another challenge for

registration algorithms. Similar challenges are faced when trying to register oblique UAV with terrestrial images, although the difference in scale between the images is not as large as in the previous scenario. This has created the need for several investigations to be carried out concerning the possibility of automatically registering images which vary in scale, viewpoint and imaging conditions.

Figure 1.1: Left: Airborne oblique image. Centre: oblique UAV image. Right: Terrestrial image.

State-of-the-art image registration methods have been developed over the years and they usually consist of three components: a *feature detector*, a *feature descriptor* and a *feature matcher*. The performance of image registration strongly relies on accurate feature detection – which is the location of salient features in an image – and robust feature description which is the encoding of information about the detected features. It is this information that's then used by an appropriate feature matcher to find corresponding features. An ideal registration method should be unique and invariant to illumination, scale, rotation and perspective (T.-Y. Yang, Lin, & Chuang, 2016). Various methods have been developed that are invariant to these differences, but research has shown that these methods may fail when these differences are exceeded beyond a certain threshold. For example, according to Geniviva, Faulring, & Salvaggio (2014), Scale Invariant Feature Transform (SIFT) (Lowe, 2004) fails in the registration of images that have a large change in viewpoint, but the improved version, Affine-SIFT (A-SIFT) compensates for this drawback to a certain extent by being able to vary the camera-axis parameters to simulate possible views making it able to account for affine viewpoints. However, due to the task of simulating all views, A-SIFT is computationally expensive and cannot simulate projective transformations (Morel & Yu, 2009). This makes SIFT and A-SIFT unreliable when it comes to the registration of images with extreme viewpoint changes, complicated geometry and large illumination variations mainly because the descriptors used are not invariant to these kind of changes.

This research aims to address the problem of automatically registering multi-resolution images, in particular, oblique UAV images to airborne oblique images since the scale variation between these pair of images is larger than the scale difference between a UAV image and a terrestrial image.

This will be done by first investigating the performance of state-of-the-art image registration methods. Afterwards, a suitable method that is invariant to differences in scale and illumination, will be modified and used to develop an algorithm fit for the application at hand. The main motive is to be able to accurately identify tie points between a pair of multi-resolution images for the photogrammetric process of relative orientation. To be more concise, the results of the research can be used to determine reliable orientation parameters of a UAV image with respect to an aerial image whose orientation is already known from direct sensor orientation. With these parameters known, subsequent UAV images of a similar scene can be integrated with other aerial images, capturing the same scene, to yield multi-resolution 3D scenes that are applicable in city planning, documentation of places of interest like cultural heritage sites, virtual tourism and so on.

1.2. Research identification

Researchers from the field of computer vision and pattern recognition have proposed a number of local invariant feature detectors (Harris & Stephens, 1988; Rosten & Drummond, 2006; Lowe, 1999) and descriptors (Alcantarilla, Bartoli, & Davison, 2012; Bay, Tuytelaars, & Van Gool, 2006; Calonder et al., 2010). These methods are well suited for various applications related to computer vision but also have a potential to be applied in the field of photogrammetry. The research aims at identifying available registration algorithms and using these algorithms to develop a procedure that is flexible enough to register multi-resolution images acquired by different imaging sensors, on different platforms, for photogrammetric applications.

1.2.1. Research objectives

The overall objective of the research is to investigate reliable methods used to register multi-resolution images with different perspectives i.e. aerial oblique and UAV oblique.

The specific objectives are:

1. Review literature and conduct experiments to evaluate the reliability of the available state-of-the-art algorithms in the registration of aerial oblique and UAV images.

2. Develop a procedure that will automatically register aerial oblique and UAV images.

3. Evaluate the performance of the developed algorithm using different image data sets that have different viewing angles and capturing a different scene.

1.2.2. Research questions

The following are the posed research questions:

1. What algorithms are available for feature detection/description for the application of registering aerial oblique and UAV images?

2. If these algorithms do exist, what are their drawbacks and can they be modified to make them more reliable in registering multi-resolution images?

3. What strategies can be utilised to develop an algorithm for the registration of multi-scale (scale range of between 2-4 times) images?

4. Which step of image registration plays a crucial role in registration process of multi-resolution images?

5. What influence does GNSS and IMU information have on the multi-scale image registration?

6. How reliable is the developed algorithm?

1.2.3. Innovation

The research aims at solving the problem of automatically registering multi-scale images for photogrammetric applications. The innovation lies in developing a registration algorithm to register images with large variations in scale. This is arrived at by; 1) Selecting a suitable feature detector/descriptor 2) Automatically determining which octaves to select in the image pair that will provide salient features for

matching 3) Selecting correct matches through multiple computations of homographies 4) Finally, combining the correspondences derived in (3) to estimate a fundamental matrix.

1.3. Thesis structure

The thesis is divided into six chapters. This chapter gives an introduction to the research by giving its motivation, research objectives and the research questions posed. Chapter two reviews several types of feature detectors, state-of-the-art feature descriptors, feature matching techniques and works related to the research topic. Chapter three embarks on the methods adopted to choose a promising feature detector/descriptor algorithm and the methods adopted to develop a procedure for multi-resolution image registration. Chapter four presents the experimental results and chapter five discusses the results. Chapter six concludes the thesis by discussing insights gained from the research and recommends future outlook in the area of study.

2. LITERATURE REVIEW

This chapter presents a brief review of the existing state-of-the-art feature detectors, descriptors and matching methods used to register images in general. These methods are compared and the advantages and disadvantages are presented. A brief review of works related to multi-resolution image registration is also presented.

2.1. Feature detectors

Feature detection is the first step in image registration, and it involves detecting features that carry crucial information about the scene captured in an image. In image registration, knowledge about corresponding points in two images is required prior the registration process. These corresponding points are actually feature points (also referred to as *interest points, keypoints, tie points* or *critical points*) and they ought to be free from noise, blurring, illumination differences and geometric differences so that similar points can be retrieved from multiple images taken of the same scene by different sensors under different environmental conditions.

Over the years, a large number of feature detectors have been developed and presented in literature. Surveys have also been done to compare and evaluate the performance of various feature detectors. Examples of such surveys include papers by Miksik & Mikolajczyk (2012), Tuytelaars & Mikolajczyk (2008), Mikolajczyk & Schmid (2005) and Fraundorfer & Bischof (2005).

This section will present a review of four common types of feature detectors that detect edge-, corner-, *blob*- and ridge-like features within an image. An overview is presented on how they work, their advantages and disadvantages, and where they are applied.

2.1.1. Edge detectors

Edge detectors employ the use of mathematical methods to identify points in an image where there is a sharp change in brightness or where there are discontinuities. These points are later fitted with lines to form edges or boundaries of regions within an image.

Canny (1986) developed a popular multi-stage algorithm to detect edges in images. The first step of the algorithm involves noise reduction because edge detection is sensitive to noise. A smoothing filter is used in this step. The next step involves calculation of intensity gradients present in the image. This is done by using a filtering kernel that computes the first derivatives in both the horizontal direction G_x and vertical direction G_y. This yields an output of two images and from these images the edge gradient and direction (given by an angle, θ) of each pixel can be computed as shown in equations 1 and 2:

$$Edge\ Gradient\ (G) = \sqrt{G_x^2 + G_y^2} \tag{1}$$

$$Angle\ (\theta) = \tan^{-1}\left(\frac{G_y}{G_x}\right) \tag{2}$$

The next step involves assigning the value zero to pixels that may not be considered to constitute an edge. This is done by checking if each pixel is a local maximum in its neighbourhood in the direction of its gradient. If a pixel does not meet this criterion, then it is not part of an edge. Otherwise, it is assigned the value of one. This eventually results in a binary image with thin lines representing plausible edges. The final step

removes edges that are not strong enough, based on a set threshold, to be referred to as edges. Two threshold values are set, a maximum value and a minimum value. All edges that have an intensity gradient above the maximum value are retained as edges whereas all edges that have an intensity value less than the minimum value are discarded. Edges whose intensity values are between these set thresholds are evaluated using a different criterion based on their connectivity. If they are connected to strong edge pixels, then they are considered to be part of the edge. Contrary to this, they are also discarded.

Another edge detector worth noting is the Sobel edge detector (Sobel, 1990). Its operation is quite similar to the canny edge detector apart from the fact that it does not make use of thresholds to retain or discard edges. This makes the detector sensitive to noise thus not as reliable as the canny detector in applications that require accurate detection of true edges.

In general, edge detectors are not suitable for some applications like image registration because the edges detected are not distinct and localised. However, edge detectors have an application in object retrieval from images for mapping purposes of line features. For instance, Ünsalan & Sirmacek, (2012) made use of the Canny edge detector to extract road networks from satellite imagery for mapping purposes. Other edge detectors implemented in the Matlab software are the Prewitt edge detector (Prewitt, 1970) and the Roberts edge detector (Roberts, 1963).

Figure 2.1 gives an illustration of the result derived after applying the Canny edge detector on an image.

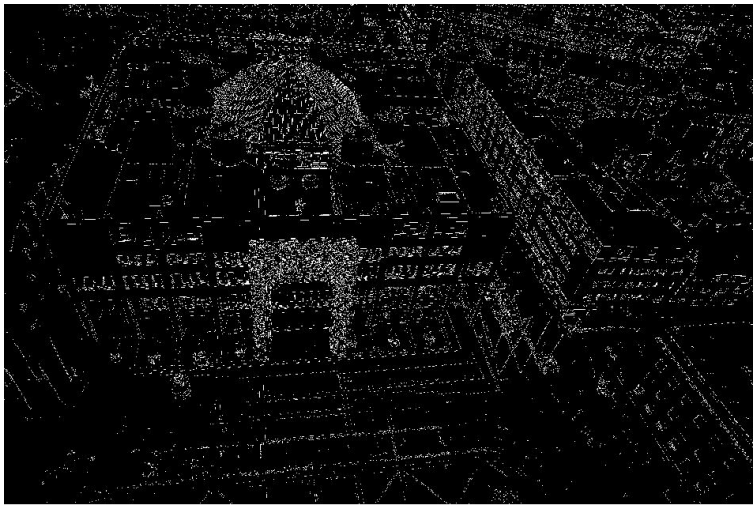


Figure 2.1: Binary image showing Canny edges.

2.1.2. Corner detectors

Corners can be defined as edge or line intersections which have large variations in image gradient in two directions. These can be considered as candidate features to detect in an image for the application of image registration because they can be localised.

Harris & Stephens (1988) developed the Harris corner detector that basically finds the intensity differences of displacements of an image patch (u, v) in all directions. This can be expressed as follows:

$$E(u,v) = \sum_{x,y} w(x,y)\ [I(x+u, y+v) - I(x,y)]^2 \tag{3}$$

w represents a filtering window which gives weights to the pixels under it. I represents the value of intensity of a pixel. In order to detect a corner, then the second term in equation 3 has to be maximized by applying the Taylor Expansion. The result can be written in matrix form as follows:

$$E(u,v) \approx [u \quad v]\, M \begin{bmatrix} u \\ v \end{bmatrix} \tag{4}$$

Where M is computed as follows:

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{bmatrix} \tag{5}$$

Where I_x and I_y are image derivatives in the x and y directions respectively. The next step is to define a criterion that aides in determining if a patch detected a corner or not. This criterion makes use of eigenvalues of the matrix M. If the first eigenvalue is higher than the second eigenvalue (or vice versa), then an edge is detected. If both eigenvalues are small, then a flat region of uniform intensity is detected. Lastly, if both eigenvalues are large and approximately equal to each other, then a corner is detected.

Another popular corner detector is the Förstner detector (Förstner & Gülch, 1987) which was developed mainly to provide a fast operator for detection and localisation of distinct points, corner and centres of circular features within an image for the application of tie point detection for photogrammetric applications. One major advantage is that the Förstner detector has the ability to detect features with a sub-pixel accuracy making it a reliable tie point detector. Contrary to the Harris detector, the Förstner detector computes the inverse of matrix M and its eigenvalues. The eigenvalues define the axes of an error ellipse. When the error ellipse is large, then a homogenous area is detected. When the error ellipse is small in one direction and large in the other direction, then an edge is detected. Lastly, when the error ellipse is small, then a corner is detected. One limitation with using the Harris and the Förstner operators is that they are not invariant to scale differences.

Additionally, FAST (Features from Accelerated Segment Test) algorithm was developed and presented in a paper by Rosten & Drummond (2006). The detector selects a pixel, p and defines a circular region around this pixel with a radius equal to three pixels. Intensity values of a subset of pixels, n within this circular region are compared to the intensity value of p plus or minus a threshold value, t. Pixel p is considered a corner if all the surrounding n pixels are brighter than $I_p + t$ or darker than $I_p - t$.

Despite being able to detect localized features, corner detectors are not invariant to scale changes of an image hence the use of region detectors which are presented in the next section.

Figure 2.2 illustrate Harris corners detected in an image.

Figure 2.2: Harris corners detected marked with green crosses.

2.1.3. Region detectors

Regions, or commonly known as *blobs,* are areas in an image that differ significantly in brightness compared to the neighbouring regions. These regions do not change under different image scales and this makes them more suitable than the earlier mentioned detectors when one needs to detect similar features between images of different scales.

The Laplacian of Gaussian (LoG) (Gonzales, Woods, & Eddins, 2014) is one of the most common *blob* detectors that first smoothens an image using a Gaussian kernel *G* (equation 6) at different scales defined by a value σ, to reduce noise and to simulate different scale levels.

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} \exp(-\frac{x^2 + y^2}{2\sigma^2}) \tag{6}$$

Then a Laplacian operator is applied to the Gaussian scale-space representation resulting in strong positive responses for dark *blobs* on light backgrounds and strong negative responses for bright *blobs* on dark backgrounds. The size of the *blobs* is directly proportional to the σ parameter.

Another method used to detect *blobs* is the Difference of Gaussians (DoG) which is an approximation of the LoG making it more efficient (Lowe, 1999). The operator makes use of subtracting a filtered image at one scale from a filtered image at a previous scale. This is done for images at different octaves[1]. Pixels of local maxima and minima are then detected in a 3×3×3 neighbourhood in the difference image as shown in Figure 2.3.

[1] Octaves are a sequence of images layered to form an image pyramid. The lowest image is the original image and the higher images are subsequently scaled down by a fixed factor.

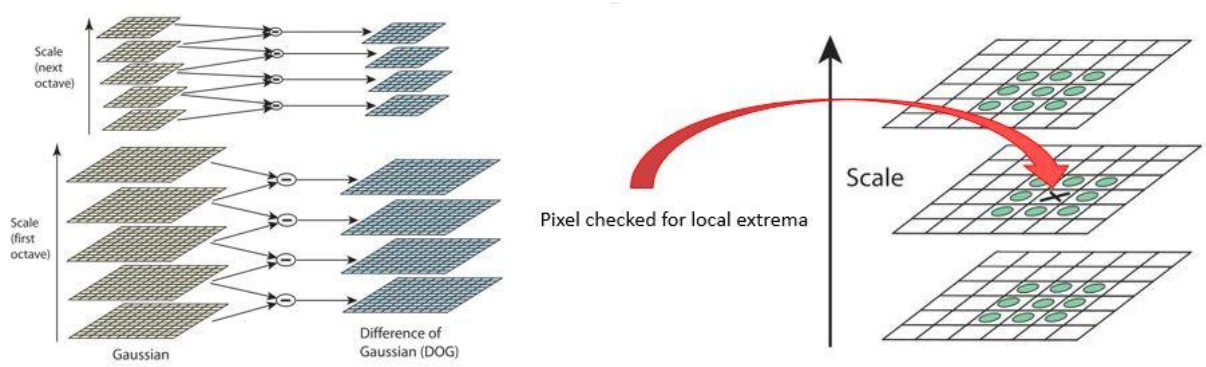


Figure 2.3: Diagram showing a representation of different image sizes (octaves) that have been smoothed by different sizes of Gaussian kernels. Difference images are obtained from adjacent filtered images and pixels o f local extrema are detected as keypoints (Lowe, 2004).

This method was implemented by Lowe and presented in his papers (Lowe, 1999, 2004). He called the detector SIFT (Scale Invariant Feature Transform).

Nevertheless, SIFT was found to be computationally expensive hence the development of SURF (Speeded Up Robust Features) (Bay et al., 2006) which uses Determinant of Hessian (DoH) to detect *blobs* in an image. The algorithm first calculates integral images and then uses box filters to smoothen the integral images which is a faster process compared to the one implemented in SIFT. Given an integral image, I and a point p with coordinates (x,y) then the Hessian matrix $H(p, \sigma)$ at point p and scale σ can be computed as follows:

$$H(p,\sigma) = \begin{bmatrix} L_{xx}(p,\sigma) & L_{xy}(p,\sigma) \\ L_{xy}(p,\sigma) & L_{yy}(p,\sigma) \end{bmatrix} \tag{7}$$

Where L_{xx}, L_{yy} and L_{xy} are the second-order derivatives of intensity with respect to the x direction, y direction and both x and y directions respectively. The determinant of this matrix is then exploited to detect stable keypoints where the determinant is maximum or minimum.

Figure 2.4 shows SURF regions detected in an image. The diameter of the circle is equivalent to the image scale and the line within the circle represents the orientation angle of the image intensity.

Figure 2.4: SURF regions detected in an image.

2.1.4. Ridge detectors

Ridges can be defined as thin lines that are darker or brighter than their surroundings contrary to edges which are discontinuities or borders between homogenous regions. The algorithm first calculates the Hessian matrix of image pixels. The eigenvalues of this matrix are then used to detect ridges if one eigenvalue is larger than the other. One typical application of using ridge detectors is in the detection of roads in Very High Resolution satellite images (Gautama, Goeman, & D'Haeyer, 2004).

2.2. Feature descriptors

After identifying distinct features in an image, it is crucial to get more information – this may be image gradients or intensity comparisons of neighbouring pixels around the centre of the detected feature – about these features and use this information to distinguish one feature from another. The description needs to be as unique and independent as possible so as to yield successful matches when finding correspondences between images of a similar scene. This description should also be robust to changes in illumination, scale, orientation and viewpoint to enable similar descriptions in other images taken of a similar scene. It is quite difficult to meet all these conditions making it needful to find a suitable trade-off.

Numerous papers have been presented over the years to evaluate the performance of descriptors. Examples include Mikolajczyk & Schmid (2005) – who compared descriptors computed for features that were scale and affine invariant – , Figat, Kornuta, & Kasprzak (2014) – who evaluated the performance of *binary descriptors* – and Krig (2014) – who gave a comprehensive survey on feature descriptors. It is evident, from these surveys, that there exists a plethora of descriptor algorithms which can be categorized into two common groups: (1) Float and (2) binary descriptors.

2.2.1. Float descriptors

They employ the use of image gradients (intensity) to describe features. The computations involved are numerous and they are done using floating digits hence the name. Normally, the image gradients of a neighbourhood of pixels around a detected feature point are computed, their orientations are assigned one of the eight possible orientations and then they are weighted. Afterwards, they are stored in a vector whose dimensions translate to the descriptor's size in bytes.

SIFT (Lowe, 1999, 2004) is the most popular float descriptor in use – also a detector as earlier mentioned – and is the benchmark used to develop other feature descriptors. It considers a 16 by 16 pixel neighbourhood

around a detected feature. Orientations of the image gradient of each of these pixels (vectors) are determined and simplified to eight possible values. These values are resolved for all pixels within a 4 by 4 array resulting in a descriptor with eight possible orientations stored in a 4 by 4 array. The descriptor vector eventually has 128 dimensions making it computationally expensive and time consuming.

Some applications such as real-time object tracking require a feature descriptor that is faster than SIFT hence the development of SURF (Bay et al., 2006) which is several times faster than SIFT because it adopts the use of Haar-wavelet response to build its descriptors. By default, instead of computing a 128 dimension feature vector it computes a 64 dimension feature vector.

SIFT and SURF are both well-known approaches in feature description but according to Pablo Fernández Alcantarilla, Nuevo, & Bartoli, (2013) they tend to suffer a drawback of not being able to preserve object boundaries by smoothening them to the same extent they do to noise at all scales. This degrades localization accuracy and robustness of features detected. To overcome this drawback KAZE features (Alcantarilla et al., 2012) were introduced and they detect and describe features in nonlinear scale spaces. This has the effect of blurring small details in the image at the same time preserving object boundaries by using a nonlinear diffusion filter. The authors claim that this method increases repeatability and distinctiveness of features as compared to SIFT and SURF but the main drawback is that it is computationally expensive and this can be attributed to the additive operator splitting (AOS) schemes that it employs to iteratively compute the nonlinear scale space.

2.2.2. Binary descriptors

Float descriptors are expensive to compute compared to binary descriptors which rely on intensity comparisons of neighbouring pixels of an interest point. These descriptors represent features as binary bit-strings stored in a vector where each digit represents the results of an intensity comparison of a pixel-pair (chosen in line with a pre-defined pattern) – which can be that a pixel is brighter or darker than the other – in an image. Immediately we can see why this family of descriptors boasts of efficiency in terms of computation and storage. Speed is fundamental in this process especially for real time and/or smart phone applications (Lee & Timmaraju, 2014).

Levi & Hassner, (2015) reviewed the design of binary descriptors and mentioned that the descriptors are generally composed of at least two parts: (1) a sampling pattern – defines a region around the keypoint for description. This can be done randomly, manually or automatically. (2) sampling pairs – identifies which pixel-pairs to consider for intensity comparison. A good example is Binary Robust Elementary Features (BRIEF) by Calonder et al. (2010) which was the first published binary descriptor. It has a random sampling pattern of point-pairs and no mechanism to compensate for an orientation of point-pairs making it a trivial method. It considers a patch of size m by m centred around a keypoint. n point-pairs (128, 256 or 512 in number) are chosen with locations (x_i, y_i) within this patch. A pair-wise comparison of intensity is computed post applying a Gaussian filter on the image to make the descriptor insensitive to noise. The comparisons are stored in binary strings ready for matching.

Another descriptor worth mentioning is Binary Robust Invariant Scalable Keypoints (BRISK) by Leutenegger, Chli, & Siegwart (2011) which uses sampling points evenly spread on a set of suitably scaled concentric circles whose sizes are directly related to the standard deviation of the Gaussian filter applied to each sampling point. This pattern is illustrated in Figure 2.5.

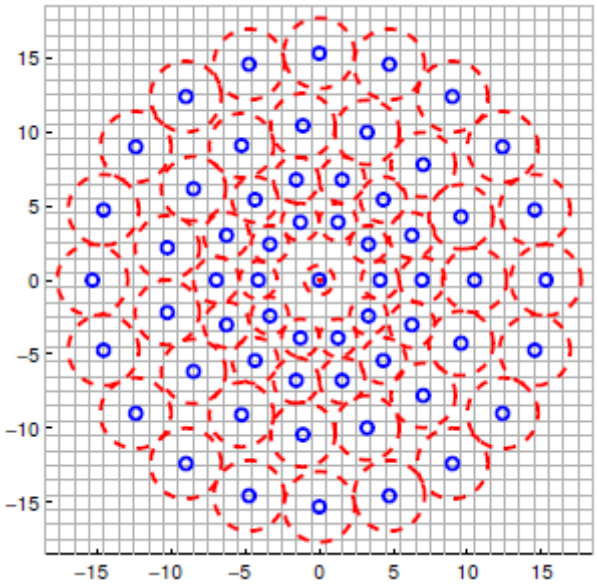


Figure 2.5: BRISK sampling pattern (Leutenegger et al., 2011)

The next step involves computing the orientation (gradient) of the sampled pixel-pairs which is implemented as follows:

$$g(p_i, p_j) = (p_i, p_j).\frac{I(p_j, \sigma_j) - I(p_i, \sigma_i)}{\|p_j - p_i\|^2} \tag{8}$$

Where $g(p_i, p_j)$ is the local gradient between a sampling pixel-pair (p_i, p_j). I is the smoothed intensity derived after applying a Gaussian filter. Subsequently, all the computed local gradients are summed up for all long pairs – a pair of sampling points that are beyond a set minimum threshold – and the overall orientation of the keypoint is calculated by solving $\arctan(g_y/g_x)$. Then the short pair – a sampling pair less than a maximum threshold – are rotated by this orientation angle to make the descriptor rotation invariant. Finally the descriptor can now be constructed by computing comparisons between a pair of short pixel-pairs using the following equation:

$$b = \begin{cases} 1, & I(p_j^{\alpha}, \sigma_j) > I(p_i^{\alpha}, \sigma_i) \\ 0, & otherwise \end{cases} \tag{9}$$

Where p_j^{α}, p_i^{α} are short pixel-pairs whose intensities are compared. If the first point in a pair has an intensity larger than the second point, then a value of 1 is assigned, otherwise, a value of 0 is assigned. The result is a string of ones and zeros and this gives the keypoint its description.

Accelerated KAZE (Alcantarilla et al., 2013) is another descriptors that makes use of binary descriptors. It's an improved version of KAZE discussed in the previous sub chapter. It uses the fast explicit diffusion (FED) (Grewenig, Weickert, & Bruhn, 2010) to speed-up feature detection in the nonlinear scale spaces. It computes descriptors based on the highly efficient Modified-Local Difference Binary (M-LDB) (X. Yang & Cheng, 2012) that exploits image gradient and intensity information from the nonlinear scale spaces making it scale invariant. Moreover, recent works by Jiang, et al. (2015) and Pieropan, et al. (2016) have demonstrated

that the AKAZE is now gaining popularity in various applications due to its performance that is rivalling other descriptors like SIFT.

2.3. Feature matching

2.3.1. Similarity Measure

In order to find corresponding features between a pair of images, an appropriate matching algorithm is required. The basic principle applied in feature matching involves comparing descriptor values with a similarity measure often referred to as *descriptor distance* (Nex & Jende, 2016). It is worth noting that this distance is not a metric distance but a similarity measure of descriptor values. The lower the descriptor distance is – below a certain threshold – between a pair of descriptors, the more likely these two descriptors are similar, hence a potential match. Various methods are used to compute descriptor distances such as L1 Norm, L2 Norm and Hamming distances. Further, the type of descriptors being matched dictates which similarity measure to use. For instance, float descriptors are compared using L1 and L2 Norm distances whereas binary descriptors are compared using Hamming distances.

Figure 2.6 illustrates the difference between L1 and L2 Norm distances.

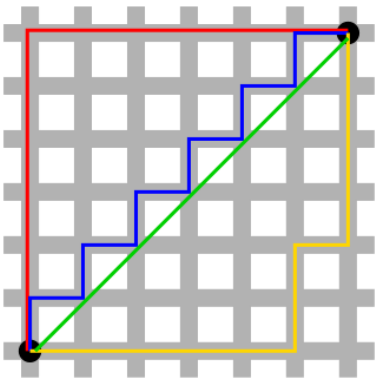


Figure 2.6: L1 Norm are coloured red, blue and yellow. L2 Norm is coloured green .

These distances are normalised and they are computed as follows:

$$|x| = \sum_{r=1}^{n} |u_r||v_r| \tag{10}$$

Where $|x|$ is the absolute distance between a pair of vectors $|u_r|$ and $|v_r|$. It is computed by summing up the lengths of line segments between two points. Figure 2.6 illustrates three possible L1 Norm distances coloured in red, blue and yellow. These are not necessarily the shortest distances hence the need for a unique shortest distance which is known as the L2 Norm distance and is computed as follows:

$$|x| = \sqrt{\sum_{r=1}^{n} |u_r|.|v_r|} \tag{11}$$

Where $|x|$ is the absolute distance between a pair of vectors $|u_r|$ and $|v_r|$. It is computed by squaring the sum of lengths between points and computing the square root of this sum.

Although equations 10 and 11 give an illustration for metric distance, the same principle is applied when computing distances between descriptor values.

On the other hand, binary descriptors are compared using the Hamming distance which is computed by performing a logical XOR operation on a pair of binary strings consequently followed by a bit count on the result. The pair of strings that has the least bit count is a potential match. This approach is faster than the former because all it requires is a binary string which has ones and zeros compared to the former which requires intensity values of pixels around a feature.

2.3.2. Matching techniques

The simplest feature matching technique is known as *brute force*. It compares the descriptor of a single feature in one image with all the other feature descriptors in the other image and returns a corresponding feature with the lowest descriptor distance.

Brute force can be efficient for a pair of images but inefficient when feature matching has to be done on a huge number of unordered images (Hartmann, Havlena, & Schindler, 2015). Projects have already been done where thousands of unordered images were implemented in a matching procedure (Agarwal et al., 2010; Frahm et al., 2010; Heinly ety al., 2015; Shan et al., 2013). Such mega projects call for a faster matcher. FLANN (Fast Library for Approximate Nearest Neighbours) based matcher offers a solution. It contains algorithms that are well suited for performing a fast nearest neighbour search for a huge dataset. This neighbourhood search can be implemented using a search structure that is, for example, based on k-dimensional trees which is a data structure that is used to organise a huge dataset of points in a k dimensional space. This strategy provides an efficient solution to find matching features.

2.3.3. Lowe's ratio test

This method implements the knn (where *k* can be replaced with an integer and *nn* stands for nearest neighbour) matching method. When k is set to, say, a value of two, then the two closest matches are returned. A threshold is then set – Lowe, (2004) suggested a threshold of 0.8. The test suggests that a corresponding match can only be considered significant if the second closest match does not share a similar descriptor distance. If that is the case, the respective descriptors are regarded as ambiguous, and that may result in a wrong correspondence. If the ratio is less than 0.8, then the match is considered to be a correct one, if this criterion is not met, then the matching pair is discarded. Reducing the threshold, reduces the number of retained matches. This method suffers a risk of discarding potentially correct matches.

2.3.4. RANSAC

As earlier stated, the resulting matches are just but mere potential matches based on descriptor distance. They are not necessarily correct matches hence the need to filter out wrong matches and actually remain with only correct matches. This is possible by using an algorithm known as RANdom SAmple Consensus (RANSAC) (Fischler & Bolles, 1981) which picks a random sample of matches and estimates the transformation between the two images based on this random sample. The matches not included in the sample are analysed to check if they are within a predefined threshold fitting the transformation model earlier estimated. This is done iteratively for a specified number of times until the highest percentage of inliers that conform to a particular transformation model is attained.

The transformation model being estimated can either be presented as a fundamental or a homography matrix.

2.3.4.1. Epipolar geometry and Fundamental matrix

The epipolar geometry is a projective geometry between a stereo pair of camera views. It's fully dependent on the cameras' intrinsics and relative orientation.

Also known as the F matrix, the fundamental matrix makes use of the epipolar geometry and the term was first coined by Luong & Faugeras, (1997). It is a 3 by 3 matrix of rank 2 which relates corresponding points in a pair of images capturing the same scene. The matrix is defined as shown in equation 12:

$$x'^{T} F\, x = 0 \tag{12}$$

Where x and x' are 3 by 1 homogenous vectors of corresponding points in the first image and the second image respectively and F is the 3 by 3 fundamental matrix with 7 degrees of freedom. A minimum of 7 corresponding image point pairs are required to solve for F. Although, there's a simpler algorithm that requires a minimum of 8 corresponding points.

According to Hartley & Zisserman, (2004), the F matrix is independent of scene structure and can be computed from corresponding image points alone without the use of camera internal parameters or relative pose. Given a pair of images that captured the same scene, each point in one image corresponds to an epipolar line in the other image. Ibid. defines the epipolar line as follows:

"The epipolar line is the projection in the second image of the ray from the point x through the camera centre C of the first camera."

From the definition of the epipolar line, there results a mapping function as shown in function 13:

$$x \rightarrow l' \tag{13}$$

Where x is a point in the first image and l' is its corresponding epipolar line in the second image. It is actually this mapping function that is exploited to constrain the search for matching features and eventually derive the F matrix.

2.3.4.2. Homography matrix

Given a pair of images capturing a planar scene, the corresponding points are related by a homography matrix (also known as the H matrix) making it scene dependent contrary to the F matrix. The relationship between these point pairs is given as follows:

$$x' = H\, x \tag{14}$$

Where x' and x are homogenous vectors of corresponding image points and H is a 3 by 3 matrix which has 8 degrees of freedom. Since H has 8 degrees of freedom, at least 4 point correspondences are required to solve H.

2.4. Related work

In relation to this research, Chen, Zhu, Huang, Hu, & Wang, (2016) proposed a new strategy for matching low-altitude (UAV) images that provided significant improvements compared to other traditional methods. The strategy was based on local region constraint and feature similarity confidence. The proposed method was compared with SIFT, Harris-Affine, Hessian-Affine, Maximally Stable Extremal Regions (MSER), Affine-SIFT, iterative SIFT and the results were convincing. The images used were oblique UAV images

captured from different viewpoints. The authors claim the method is efficient but it highly depends on the image content meaning it works better for images that captured structured scenes.

Geniviva et al. (2014) proposed an automated registration technique that could be used to improve the positional accuracy of oblique UAV images using orthorectified imagery. The technique implemented the A-SIFT algorithm to find correspondences between the oblique UAV images and orthorectified imagery. A-SIFT was used due to its ability to vary the camera-axis parameters in order to simulate all possible views. However, the algorithm used is computationally expensive and it does not account for projective transformations.

Koch et al. (2016) proposed a new method to register nadir UAV images and nadir aerial images. An investigation was done to assess the viability of using SIFT and A-SIFT. It was concluded that these methods failed due to the fact that the images to be matched had a large difference in scale, rotation and temporal changes of the scene. This led to the proposed method which used a novel feature point detector, SIFT descriptors, a one-to-many matching strategy and a geometric verification of the likely matches using pixel-distance histograms. The reliability of this method to register aerial oblique to UAV oblique images was not investigated.

Jende et al. (2016) proposed a novel approach for the registration of Mobile Mapping (MM) images with high-resolution aerial nadir images. The approach involved using a modified version of the Förstner operator to detect feature keypoints only in the aerial ortho-image. The feature keypoints are then back projected into the MM images. A template matching strategy is used to find correspondences as opposed to using feature descriptors. The approach was compared to AGAST detector & SURF descriptor and Förstner detector & SURF descriptor. The reliability of this method to register aerial oblique to UAV images was not investigated.

Gerke et al. (2016) performed experiments to investigate on how current state-of-the-art image matching algorithms perform on terrestrial and UAV based images. They also investigated the role played by image pre-processing on the performance of the algorithms. However, tests on airborne images were not performed.

Most of the previously mentioned research do not give a solution to register airborne oblique to UAV images hence the emphasis on this research.

3. METHODS AND MATERIALS

This chapter gives a detailed explanation of the methods, datasets and tools used to choose a promising image matching algorithm, and the experiments conducted that led to tailoring the chosen algorithm to register the image pairs that this research is interested in. Figure 3.1 shows a general overview of the work flow implemented to develop the algorithm.

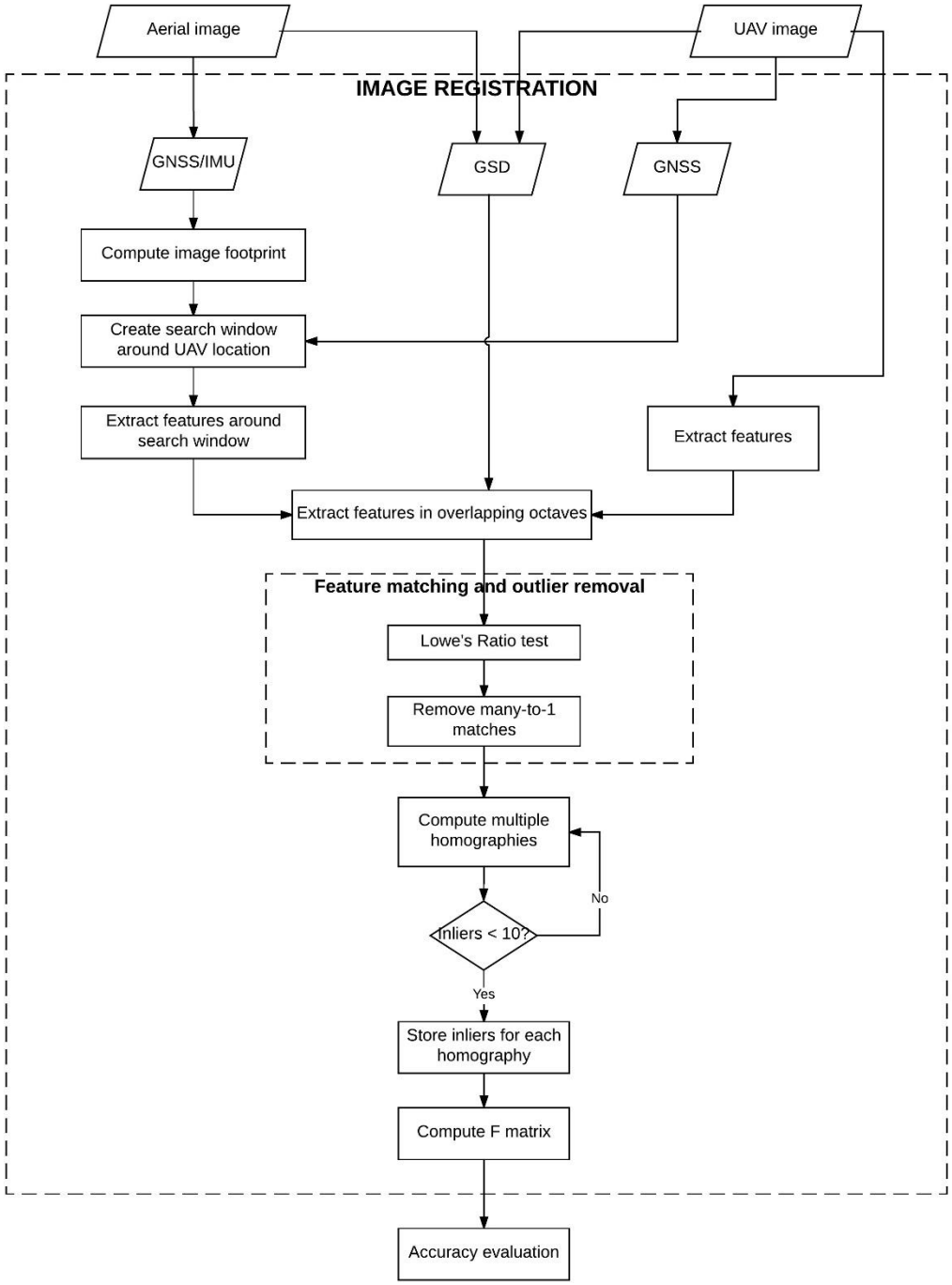


Figure 3.1: General overview of the methodology adopted for registering aerial oblique and UAV images.

3.1. Algorithm selection

After performing a literature review on the various image matching algorithms, six algorithms were selected depending on the type of features detected – scale invariant – and the feature descriptors. The image pair in Figure 3.3 (a) (on page 23) was chosen to test the algorithms since it didn't have an additional challenge of viewing angle differences compared to the other image pairs. Looking at the challenge evident in Figure 3.3 (a), the resultant algorithm ought to be invariant to scale differences. This was ensured by choosing scale invariant detectors and leaving out edge, corner and ridge detectors. When it came to choosing descriptors, a fair selection was done to select three float descriptors and three binary descriptors. This led to the selection of SIFT, SURF, KAZE, SURF/BRIEF, BRISK and AKAZE. These algorithms were tested using their default settings.

A general pipeline was implemented where the first step involved detection and description – also known as feature extraction – of salient features within the image at different scales. This was followed by matching the descriptors so as to find corresponding points between the image pair. Apparently not all matches were absolutely correct hence the need to remove outliers by using RANSAC. Finally, the inliers were visually checked for correctness to determine the reliability of the image matching algorithm.

The following sub sections describe the default parameters that were implemented for each of the six chosen algorithms.

3.1.1. Feature extraction

Table 3.1 gives the default parameter settings used to test SIFT, SURF, KAZE, SURF/BRIEF, BRISK and AKAZE.

Table 3.1: Default parameter of the chosen feature detector/descriptor

Algorithm	Parameters					
	No. of octaves	Contrast threshold	Edge threshold	sigma	Hessian threshold[2]	Descriptor size
SIFT	-	0.04	10	1.6	-	128
SURF	4	-	-	-	100	64
KAZE	4	-	-	-	0.001	64
SURF/BRIEF	4	-	-	-	100	32
BRISK	3	-	-	-	30	64
AKAZE	4	-	-	-	0.001	64

SIFT doesn't allow the user to adjust the number of octaves. This is done automatically depending on the image resolution. The contrast threshold is used to filter out weak features in image regions of low contrast. Increasing the value reduces the number of features detected. Contrary to what the edge threshold does, where a larger value retains more features. The *sigma* represents a parameter used in the Gaussian filter applied to the image to introduce a blurring effect that reduces image noise. The Gaussian filter is given by equation 15.

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \tag{15}$$

[2] Partial derivatives of image intensities around a pixel are used to build an approximation of the Hessian matrix. The determinant of the matrix is what is referred to as the Hessian. Setting a threshold determines from which value will keypoints be detected.

Where *x* and *y* are pixel positions in the image.

As for SURF, the number of octaves can be altered and the default is set to four. This means that the original image is downsampled by a factor of two, successively, until an image pyramid with four images is formed. Increasing the number of octaves, results in detection of large features and vice versa. Features larger than the Hessian threshold are retained. Increasing the value results to less features being detected and vice versa. Finally, the feature descriptor has a default size of 64 compared to SIFT which has a size of 128 dimensions.

The number of octaves used in KAZE is similar to SURF. The same applies for its descriptor size. Its Hessian threshold value of 0.001 plays the role of retaining features. Increasing the value will result to less features being detected and vice versa.

The BRIEF descriptor does not come with its own detector. Therefore, an arbitrary choice had to be made for a detector that's scale invariant, hence SURF due to its efficiency in feature detection compared SIFT and KAZE. The only noteworthy parameter available for the BRIEF descriptor is the length of the descriptor which is 32 bytes by default and plays a role of easing computations when it comes to matching its descriptors.

The third algorithm uses FAST to detect features that are beyond a threshold of 30, in a default number of three octaves. The BRISK descriptor, with a size of 64 bytes, is employed.

Finally, AKAZE uses a similar number of octaves as SURF and KAZE. The threshold default value is 0.001, similar to KAZE and it plays a similar role of retaining features.

3.1.2. Matching the descriptors

Float descriptors were matched using brute force based on Euclidean distance while binary descriptors were matched using brute force hamming distance. Thereafter, Lowe's ratio test was implemented to discard mismatches. A final screening was done to check for many-to-1 matches. In case any were found, then they were removed but retaining the one with the least distance.

3.1.3. Outlier removal

RANSAC was used to remove the outliers by estimating a fundamental matrix. The default parameters that were used are: 1) Inlier threshold of 0.001 2) Minimum number of eight sample points.

The number of trials is dependent on the confidence level set by the user and the number of putative matches. Equations 16 and 17 (Mathworks, 2012) show how the number of trials is determined for each iteration run.

$$N = \min(N, \frac{\log(1-p)}{\log(1-r^8)}) \tag{16}$$

Where *p* represents the confidence parameter set by the user and *r* is calculated as shown in equation 17.

$$r = \sum_{i=1}^{N} sgn(d\ u_i, v_i\ ), t)\ /N \tag{17}$$

Where *sgn(a,b)* = 1 if $a \leq b$ and 0 otherwise.

3.2. Reduction of search area

In order to improve the results in the matching step, it was deemed necessary to restrict the search area for matching features within the area of overlap in the aerial image. The available internal and external camera parameters for both images were exploited to achieve this objective. On the one hand, the aerial images came with GNSS and IMU information which offered approximate values for exterior orientation (EO). The camera used was calibrated and this means that crucial information about its parameters were availed in its camera calibration report. On the other hand, the UAV images had GNSS information embedded in their respective Exchangeable Image File (EXIF) tags together with basic camera parameters like the focal length, image resolution and pixel size. A piece of information missing conspicuously, is the orientation of the images which was not offered by the vendor, possibly due to the UAV payload capacity not being able to host an IMU on board. Notwithstanding, an oblique UAV image, with a viewing angle approximately equal to that of the aerial image was chosen. This was discerned by careful visual inspection.

Figure 3.2 shows a sketch of the geometry between the aerial and UAV camera. This configuration assumes that the UAV's viewing angle was similar to the one adopted by the aerial camera.

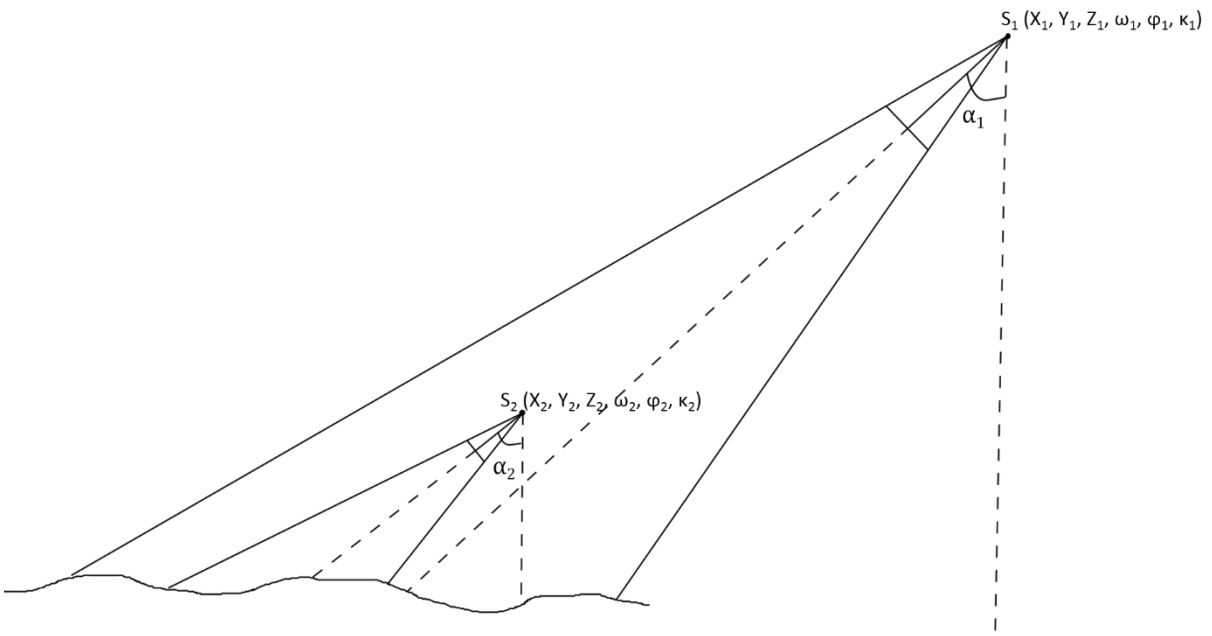


Figure 3.2: Geometry of the aerial and UAV camera. S_1 represents the position and orientation of the aerial camera recorded by on board GNSS and IMU. S_2 represents the position of the UAV camera recorded by an on board GNSS. α_1 and α_2 represents the tilt angle of the respective cameras (Figure not drawn to scale).

With all the information at hand, the position of the UAV was located on the aerial image. This was done by first projecting the four corners of the aerial image plane to determine their world coordinates. The collinearity equations 18 and 19 were implemented to achieve this.

$$X = Z - Z_o \frac{R_{11}x + R_{21}y - R_{31}c}{R_{13}x + R_{23}y - R_{33}c} \tag{18}$$

$$Y = Z - Z_o \frac{R_{12}x + R_{22}y - R_{32}c}{R_{13}x + R_{23}y - R_{33}c} \tag{19}$$

Where x and y represent the image coordinate of a corner of the aerial image plane, R_{11} to R_{33} are elements of the rotation matrix, c is the camera focal length, X, Y and Z (average terrain height of the area captured by the aerial image) are the ground coordinates of x,y. Z_o is the height of the camera at the instant of image capture.

The next step was to determine if the UAV image coordinates were actually within the four corners in the ground reference system. If this was the case, then the UAV image coordinates were back projected to the aerial image plane using equations 20 and 21.

$$x = -c\frac{R_{11}(X - X_O) + R_{12}(Y - Y_O) - R_{13}(Z - Z_o)}{R_{31}(X - X_O) + R_{32}(Y - Y_O) - R_{33}(Z - Z_o)} \tag{20}$$

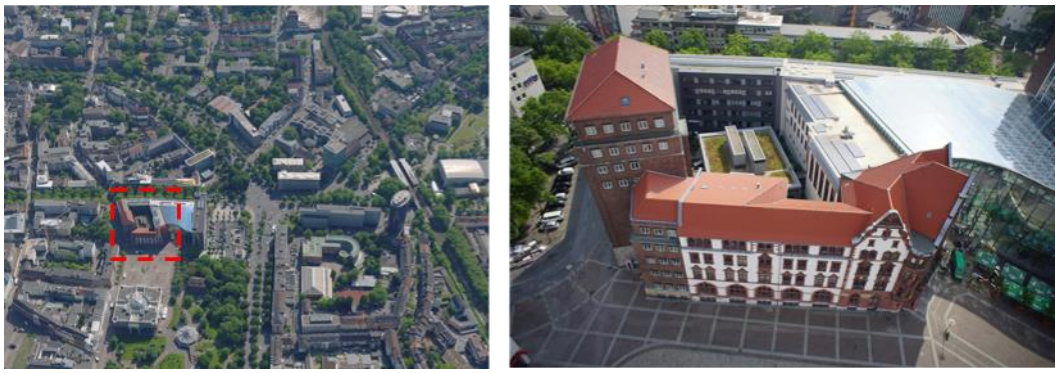
$$y = -c\frac{R_{21}(X - X_O) + R_{22}(Y - Y_O) - R_{23}(Z - Z_o)}{R_{31}(X - X_O) + R_{32}(Y - Y_O) - R_{33}(Z - Z_o)} \tag{21}$$

Where x and y represent the image coordinate of UAV on the aerial image plane, R_{11} to R_{33} are elements of the rotation matrix, c is the camera focal length, X, Y and Z are the ground coordinates of the UAV at the instant of image capture and X_o, Y_o and Z_o are the ground coordinates of the aerial camera at the moment of image capture.

The back projected point is now an approximate image location of the overlap area. Thereafter a bounding box of 1000 by 1000 pixels around the image is chosen to represent the restricted search area for corresponding features to match with. This window size was chosen because features were easily discernible in the aerial image within this window.

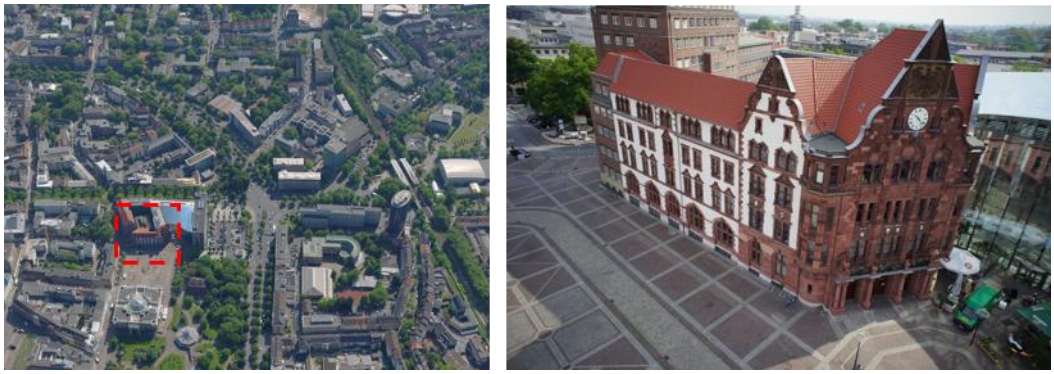
3.3. Image pair selection

Four image pairs – aerial and UAV images – were chosen for two different buildings. Since the images are taken from different platforms, flying at different heights, they have different scales and this is the main challenge this research is trying to overcome. The chosen image pairs are shown in Figure 3.3 (a), (b), (c) and (d). Figure 3.3 (a) shows images that seem to have been taken from a similar viewing angle and the illumination differences are not outstanding. In Figure 3.3 (b), the viewing angle difference between the aerial camera and the UAV camera is slightly different from the one adopted in Figure 3.3 (a). The UAV almost had a horizontal viewing angle to the building. Figure 3.3 (c) has a UAV image that was taken from a side-looking view of the building. Finally, Figure 3.3 (d) captured a different scene with both images taken with cameras having approximately similar viewing angles. These different pairs were chosen to evaluate the performance of the algorithm under different scenarios.



(a): Pair 1

(b): Pair 2



(c): Pair 3

(d): Pair 4

Figure 3.3: (a)-(c) Left: aerial oblique image. Right: UAV image of Stadthaus in Dortmund city centre. (d) Left: Aerial oblique image. Right: UAV image of Rathaus in Dortmund city centre. The dashed red box in the left images represent the overlapping area of the respective image pairs.

3.4. Experimental study

The various parameters of AKAZE were tuned in the quest of finding suitable settings that will actually result in achieving acceptable image registration results. This sub chapter explains the methods adopted in improving the results obtained by using AKAZE and design choices that were made to develop an algorithm to suit the application of this research.

3.4.1. Feature detection and description

Two main parameters that are relevant to this research are the Hessian threshold and number of octaves. These parameters were adjusted accordingly, making use of information – like image GSD – about the images to be registered. The effect of these parameters were investigated independently to determine the significant role played by each of them in the registration process.

Hessian threshold

This threshold determines the number of detected features in the image. When detecting AKAZE features, a default number of 0.001 (OpenCV, 2012) is used. Figure 3.4 shows an illustration of how the number of detected features decay from lower to higher octaves. The same applies for UAV images which actually needed more features to be detected in the higher octaves. This led to reducing the threshold to 0.0001. The number was further reduced to 0.00001 so as to detect more salient features in the higher octaves of the UAV image.

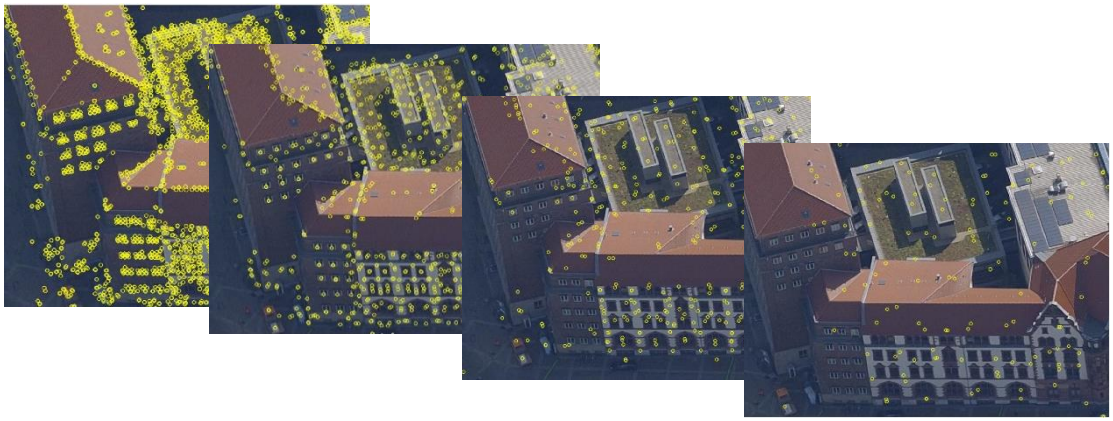


Figure 3.4: Detected features in the four octaves of an aerial image.

Octaves

This is the parameter that actually makes the algorithm scale invariant by creating an image pyramid with the original image at the base of the pyramid and images subsequently downsampled by a factor of 2, stacked in a hierarchical manner up the pyramid. The number selected, determines the number of images in the pyramid. The default is four. An investigation was done to identify exactly which features, from which octaves, actually matched and to determine which octave pairs to select. This was necessary because it is unlikely to find matching points between the lowest octaves of the two images due to the huge scale differences and it is expected to find matching features between the lower octaves of the aerial image and the higher octaves of the UAV image because the octaves resemble in scale, more or less. A table showing the results of this investigation are presented in the next chapter.

The average GSD of the images used was also computed (see equation 22) to determine their relationship with the overlapping octaves. To compute the GSD, the flying height of the platforms, the focal length of the respective cameras, the cameras' angle of tilt and the pixel size of the respective image planes were required before hand. Most of this information is availed within the EXIF tags of the images. The exterior and interior orientation parameters of the aerial image was availed whereas the angle of tilt of the UAV was

assumed to be similar with the one adopted by the aerial camera. This relationship aided in automating the image registration process by choosing octaves in the UAV image that match with an octave in the aerial image.

$$GSD = \frac{H}{f} * pix \tag{22}$$

Where H is the flying height of the platform above ground level, f is the focal length of the camera and pix is the pixel size of the camera frame. All parameters are in millimetres.

Descriptor size

The descriptor extracts information of neighbouring pixels around the feature and stores their intensity values as binary values which is 64 dimensions for AKAZE. This default value was maintained.

3.4.2. Feature matching criteria

Different methods were used to find corresponding features. They are presented as follows:

Brute force matching

This method utilises an exhaustive search procedure where every feature in the UAV image was compared with every feature in the aerial image. Features with the highest similarity measure, i.e. lowest distance between them, were returned as putative matches. When using only this method in descriptor matching, it suffers the risk of returning many-to-one matches and other wrong matches hence the use of Lowe's ratio test.

Removal of many-to-one matches

To counter the potential drawback that might be suffered by brute force matching, it is important to have unique matches – i.e. one-to-one matches – at the end of the matching procedure. This gives ground for actually relying on these matches. Hence the need to filter out one-to-many matches.

This was done by first identifying the many-to-one matches and analysing their respective distances. The matches with larger distances were discarded whereas the match with the smallest distance was retained as potential match.

3.4.3. Multiple homographies

Now that putative matches had already been computed, the next step was to estimate the fundamental matrix – geometric relationship between the image-pair – using RANSAC which requires approximately 50-60 percent outliers so as to have a reliable matrix. Since the putative matches were not all correct matches, it was decided to employ the computation of multiple homographies so as to filter out wrong matches in every iteration. Zuliani, Kenney, & Manjunath, (2005) used a similar approach and they called it multiransac.

The computation of a homography between a pair of images is dependent on planar elements in a scene. This research uses images of buildings. The buildings have structured surfaces with varying shapes and orientations making them multi-planar. Figure 3.5 illustrates this concept.

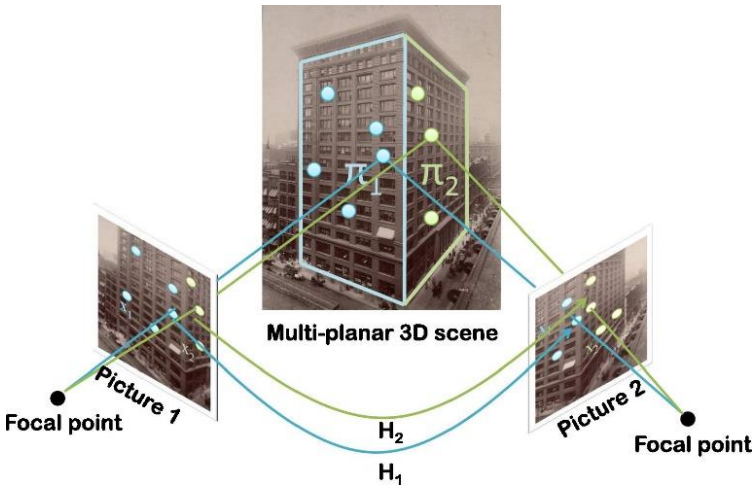


Figure 3.5: A building scene represented as having two planes. Homologous points from each plane have a homography mapping (Szpak et al., 2014).

With this hypothesis in mind, multiple homographies were computed iteratively using the putative matches, earlier computed, as the only input.

As mentioned in chapter two, to derive a homography, at least four point pairs are required. RANSAC was used to look for points conforming to a homography. The first iteration takes the whole set of putative matches and computes the first homography. The inliers are stored while the outliers are used in the next iteration to compute the second homography. This is done iteratively until no more inliers are found. A condition was set to stop the iteration whenever less than ten points were detected to compute a homography.

3.4.4. Fundamental matrix

After computing multiple homographies, a considerable amount of outliers were filtered out. The next step was to compute a global geometric relationship that exists between the image pairs. This was done by estimating a fundamental matrix using the inliers that were stored for every homography computed previously.

The eight point algorithm (Longuet-Higgins, 1981) was used to compute the F matrix due to its simplicity as compared to the seven point algorithm which has the disadvantage of potentially giving three possible solutions, all of which must be tested (Hartley & Zisserman, 2004). The algorithm picks a random sample of eight correspondences, determines a model – which is the F matrix – and looks for other correspondences that fit to this model. The higher the number correspondences throughout the scene, the higher the chances of deriving an accurate F matrix.

3.4.5. Performance and accuracy evaluation

The algorithm was tested for three other different scenarios; two different viewing angles of Stadthaus and a different building in Dortmund (Rathaus) as shown in Figure 3.3 (b)-(d).

Accuracy evaluation was done by making use of the epipolar constraint discussed in chapter one. Corresponding epipolar lines were computed and metric distances from these lines to their corresponding points were derived. An ideal case will result to distances equalling to zero but in reality this might not always be the case possibly due to localisation errors encountered during feature detection. Computing the average residual error, as shown in equation 23, was used to assess the accuracy of the F matrix in mapping point features from the aerial image to corresponding epipolar lines on the UAV image. The average residual error

was compared with the average residual error computed from manually identified matches throughout the scene.

$$\frac{1}{N}\sum_{i}^{N} d'(x_i', F\hat{x}_i) \tag{23}$$

Where N is the total number of matching points and $d(x, Fx)$ is the distance between a point to its corresponding epipolar line on the other image. Figure 3.6 depicts this relationship. Image 1 can be used to represent the aerial image while image 2, the UAV image.

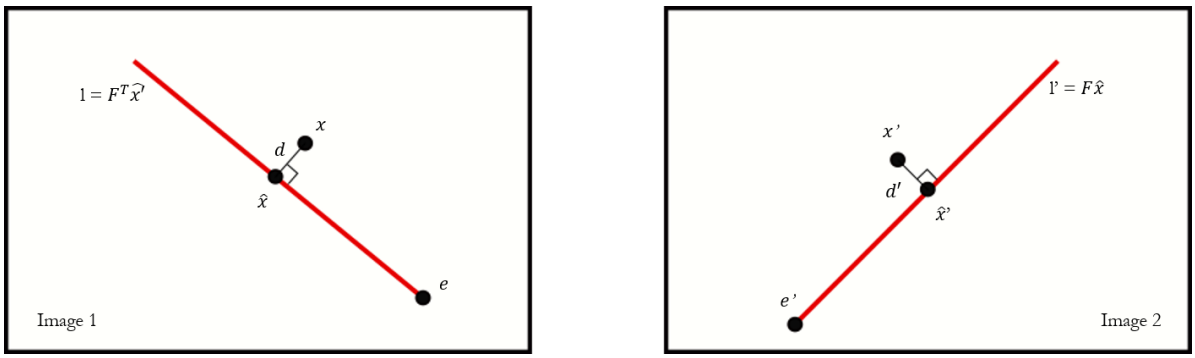


Figure 3.6: Relationship between epipolar lines and corresponding points.

Since the objective is to register the UAV image to the aerial image, then residual error between matched points was computed on the UAV image.

3.5. Auxilliary test

An additional test was performed to improve the number of matches. A pair of images might have been taken at different dates, hence the possibility of different illumination conditions. This gave enough reason to conduct further tests to figure out a method to successfully register multi-scale images captured under these varying conditions. The method used to conduct this additional test is discussed as follows:

Wallis filter

So as to reduce the illumination differences between a pair of images, Wallis filter (Wallis, 1979) was applied on both images as a pre-processing step. Jazayeri & Fraser, (2008) reported that by applying the Wallis filter, issues arising from illumination are overcome leading to more repeatable and reliable detected features. To add, Gerke et al. (2016) also reported to have improved the matching results for a pair of images with varying contrast after applying the Wallis filter and using the SIFT algorithm.

It's worth mentioning how the Wallis filter works. It is a locally adaptive filter that enhances the contrast of a grayscale image with significant areas of bright and dark tones. Contrary to global filters, this filter provides an even contrast throughout the image thus eliminating the variations in illumination. It was of interest to find out how AKAZE responded to Wallis filtered images.

3.6. Dataset and software

The experiments were conducted using image data sets provided to scientific researchers in the framework of the multi-platform photogrammetry benchmark (Nex et al., 2015) undertaken by ISPRS and EuroSDR

scientific initiative. The data sets used comprise of aerial oblique images (of Dortmund) and UAV images of Stadthaus and Rathaus buildings in Dortmund's city centre. The aerial oblique images had GNSS and IMU information and the UAV images had GNSS information encoded in their EXIF tags.

The tests were conducted in Matlab using integrated OpenCV (Open Computer Vision) functions together with built-in Matlab functions.

4. RESULTS

This chapter presents the results and discussions of the preliminary and subsequent tests that led to answering the research questions of this research. Some of the figures derived from the experiments are presented herein, but more figures can be found in the appendices section.

4.1. Algorithm selection

Six different algorithms – SIFT, SURF, KAZE, SURF/BRIEF and BRISK – were tested using their default settings to match the pair of images shown in Figure 3.3 (a) (page 22). These algorithms were chosen so as to have a fair share of both float and binary descriptors, and since this research has a focus on registering multi-scale images, then all the chosen algorithms employed scale invariant feature detectors.

Two tests were performed to assess the performance of the chosen algorithms. The first test involved matching the image pair without restricting a search area in the aerial image while the second test involved cropping the overlapping area in the aerial image. Figure 4.1 illustrates an analysis of the matching results achieved for each of the six chosen algorithms, after performing the first test.

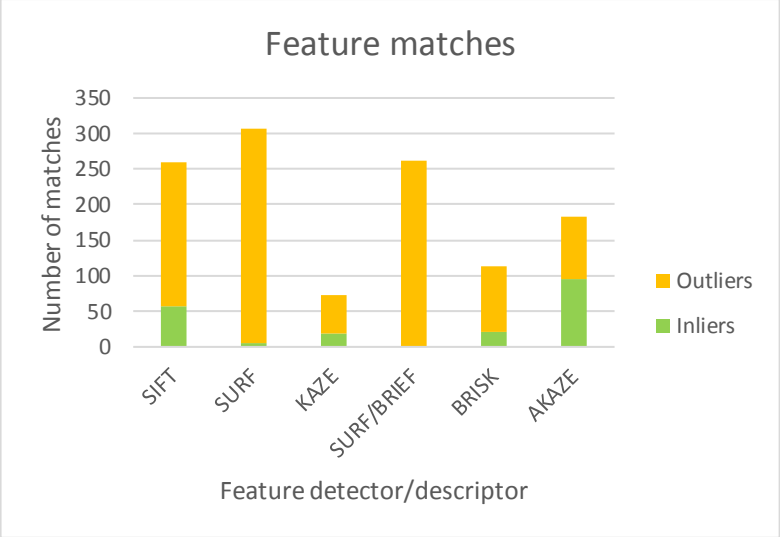


Figure 4.1: Analysis of feature matching results between different detector/descriptors for an uncropped aerial image and a UAV image as shown in Figure 3.3 (a) (page 22).

From Figure 4.1, it is clear that AKAZE outperformed SIFT, SURF, KAZE, SURF/BRIEF and BRISK by being able to detect close to 100 correct matches (slightly more than 50 percent of the total putative matches). The other algorithms detected less than 25 correct matches, with SURF/BRISK barely detecting a correct match, and SIFT detecting just over 50 correct matches (but less than 50 percent of its total putative matches). These results are interesting, because AKAZE uses binary descriptors that have been reported to perform poorly compared to float descriptors (Trzcinski & Lepetit, 2012).

Figure 4.2 shows the matching results achieved by AKAZE before cropping the region of overlap in the aerial image. The other results are shown in appendix 1.

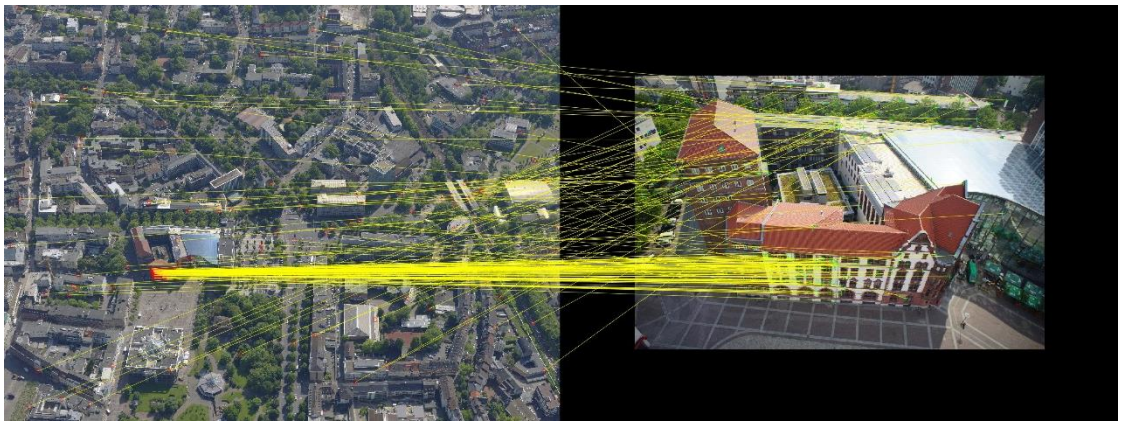


Figure 4.2: AKAZE matches between an uncropped aerial image and a UAV image.

Despite being able to achieve the highest number of correct matches, it was evident that a significant amount of features outside the area of overlap of the aerial image were mismatched as shown in Figure 4.2. This can be due to the repetitive nature of trees and vegetation captured in both images, a phenomenon that causes a drawback in image matching.

So as to improve the chances of acquiring more correct matches, a second test was performed to assess the impact of automatically restricting the search area for matches in the aerial image. This design procedure led to an improvement of the results as shown in Figure 4.3.

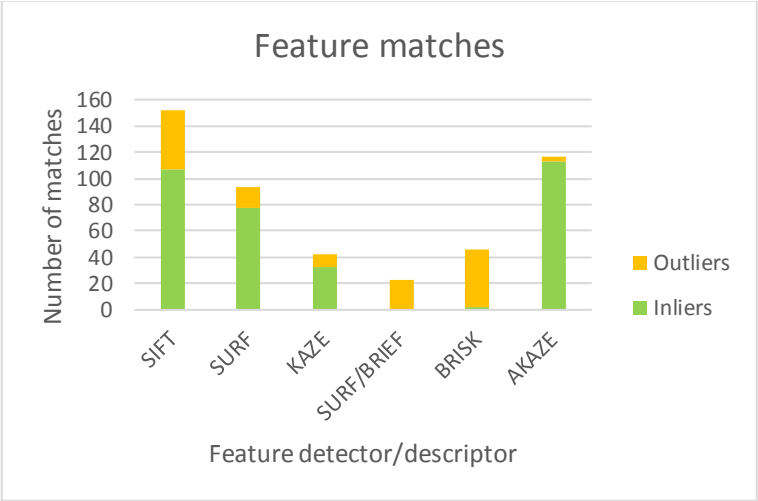


Figure 4.3: Analysis of feature matching results between different detector/descriptors for a cropped aerial image and a UAV image.

The results indicate an improved performance in the algorithms that use float descriptors which were all able to detect more than 50 percent correct matches, while SURF/BRIEF and BRISK performed poorly, compared to AKAZE, which still outperformed all the other five algorithms by managing to compute almost 100 percent correct matches. Figure 4.4 illustrates the matches computed with AKAZE (The figures illustrating the other results are found in appendix 2). It is clear that a large percentage of the putative matches were mostly detected on the façade of the building and a small percentage of the putative matches were detected elsewhere within the scene. This distribution of matches can be attributed to the fact that the building façade has a good texture and varying contrast for matching hence distinct features are easily detected in this part of the image. Another reason might be because the features on the building façade were detected in both images by the feature detector whereas features from the other parts of the image were not detected in the both images.

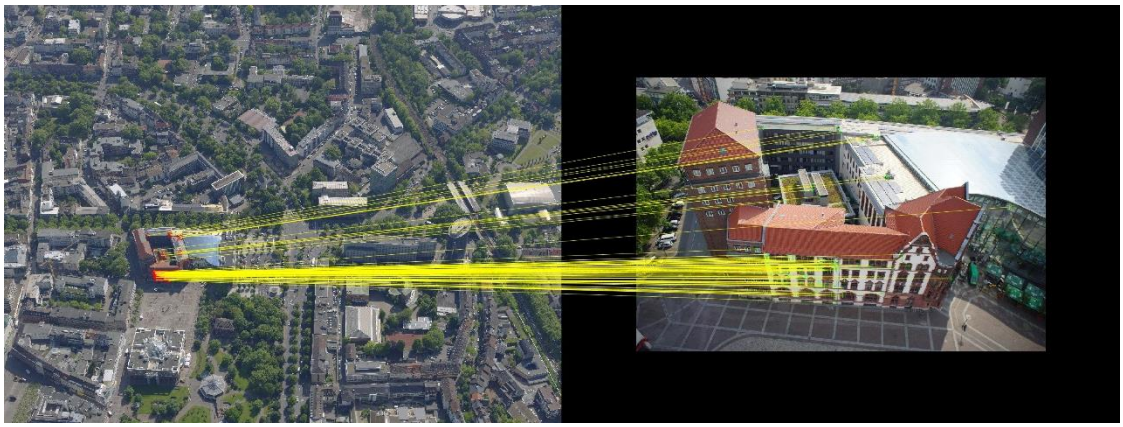


Figure 4.4: AKAZE matches between a cropped aerial image and a UAV image.

It is highly suspected that AKAZE outperformed the other algorithms due to its nature of preserving boundary features as opposed to, say SURF and SIFT, that erase boundary features when denoising the image during the filtering step. This results to AKAZE detecting salient features along window edges, roof edges etc. Although KAZE works in a similar way as AKAZE during feature detection, both use different schemes in feature detection and work differently during feature description and this could be the reason why AKAZE outperformed KAZE. AKAZE uses an FED scheme compared to KAZE that uses an AOS scheme to detect features. The former is more accurate when it comes to localisation of features. In addition, the AKAZE descriptor employs the use of a modified version of the Local Binary Descriptor that has been reported by Alcantarilla et al. (2013) to be highly efficient and invariant to scale changes.

Main findings

- AKAZE outperforms SIFT, SURF, KAZE, SURF/BRIEF and BRISK in both experiments hence the reason why it was selected.

- Restricting the search area for matches in the aerial image generally improves the results of the feature matching algorithms significantly and this process can be automated by exploiting selected camera EO parameters of both images.

- Repetitive features such as trees and vegetation may lead to errors in image matching while good textured surfaces provide robust features suitable for accurate matching.

4.2. Impact of tuning feature detection parameters

4.2.1. Octaves

Table 4.1 shows the number of putatively matched features detected in various octaves of the aerial and UAV images.

Table 4.1: Analysis of octaves that produced putatively matched keypoints.

	Aerial image	UAV image	Matches
octaves	1	3	54
	2	4	37
	1	4	19
	1	2	5
	3	4	2

Most correct matches were detected between the first octave of the aerial image and the third octave of the UAV image, followed by octaves two and four, one and four, respectively. Fewer matches were detected between the other octaves as shown in Table 4.1 and no matches were detected in the other octave combinations (not shown in Table 4.1).

The results presented in Table 4.1 show that potential features for matching are found in the lower octaves of the aerial image and the higher octaves of the UAV image. This indicates that there is a relationship between matching features and scale. This led to the use of image GSD to automatically extract only features from octaves that are more likely to yield favourable candidates for matching. Table 4.2 shows the GSD at the centre of the respective images and the parameters used in the GSD computation.

Table 4.2: GSD between aerial and UAV images

Imaging platform	Pixel size (μm)	Focal length (mm)	Flying height (m)	Angle of tilt (degrees)	GSD (cm/pixel)[3]
Aerial	6	80	1033.78	45	10.96
UAV	3.9	16	57.72	45	1.99

The computed GSD values show that there's a ratio of approximately 5.51 between the image pair. This ratio was the condition used to select octaves 1 and 2 of the aerial image, and octaves 3 and 4 of the UAV image respectively.

4.2.2. Feature detection threshold

From Figure 4.4 it can be seen that the distribution of correct matches is uneven throughout the image. This can be due to the fact that few or no distinct features were detected in the higher octaves of the UAV image, and lower octaves of the aerial image. Figure 4.5 gives an illustration of how feature detection decays from low to high octaves. It is also clear that feature detection decays from the high resolution image to the low resolution image. The UAV image captured more details – since it was captured at a lower flying height than the aerial image – compared to the aerial images hence the observation.

[3] The GSD at the principal point on the images is considered.

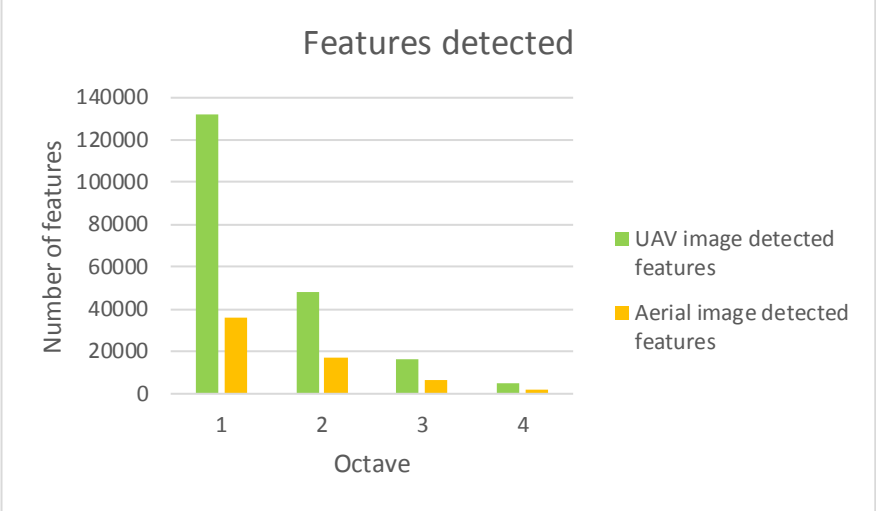


Figure 4.5: Analysis of the number of features detected in the four octaves of the UAV and aerial image.

Additionally, feature detection threshold plays a crucial role in determining the number of features detected in an image. The value – a default of 0.001 – is inversely proportional to the number of detected features. Figure 4.6 shows a comparison between the features detected with a threshold of 0.001 and 0.0001. The number of features increased and their distribution improved. More features were detected in the lower left and middle parts, as shown in the right image, compared to the left image, where no features were detected in similar parts of the image.

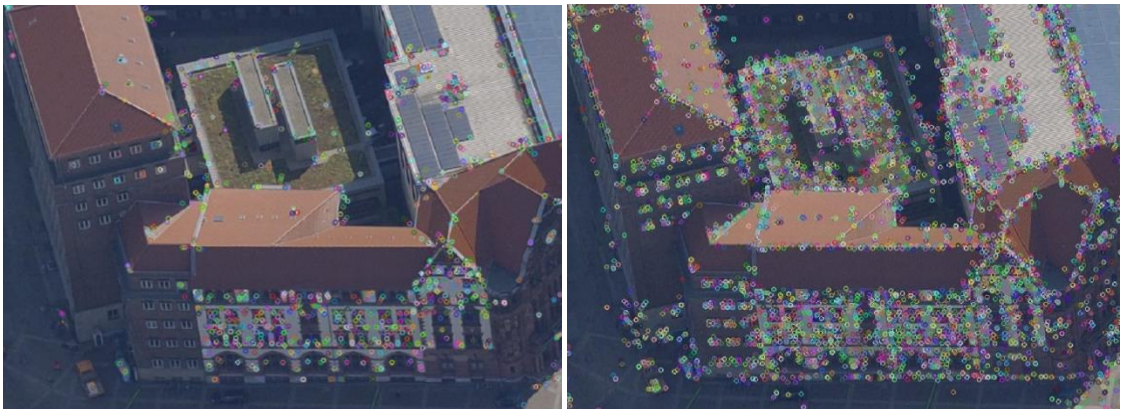


Figure 4.6: Aerial image of Stadthaus showing partially detected features (left) and evenly detected features (right).

An increase in the number of detected features in the higher octaves led to an increase in the number of matches. This is because more salient points were detected hence increasing the chances for successful matches compared to when fewer salient points were detected in the higher octaves of the UAV image. Figure 4.7 shows the matching results analysis achieved after lowering the threshold for feature detection in both images.

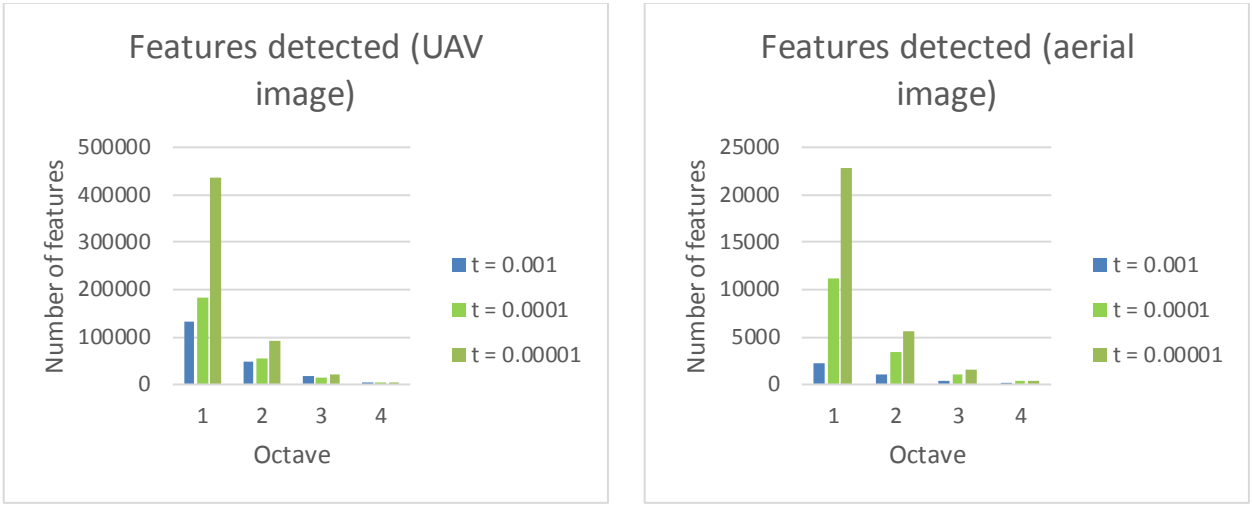
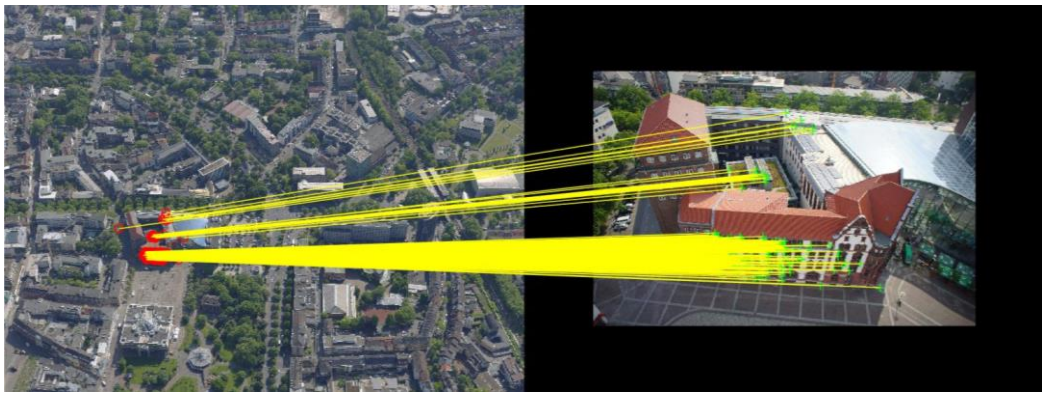
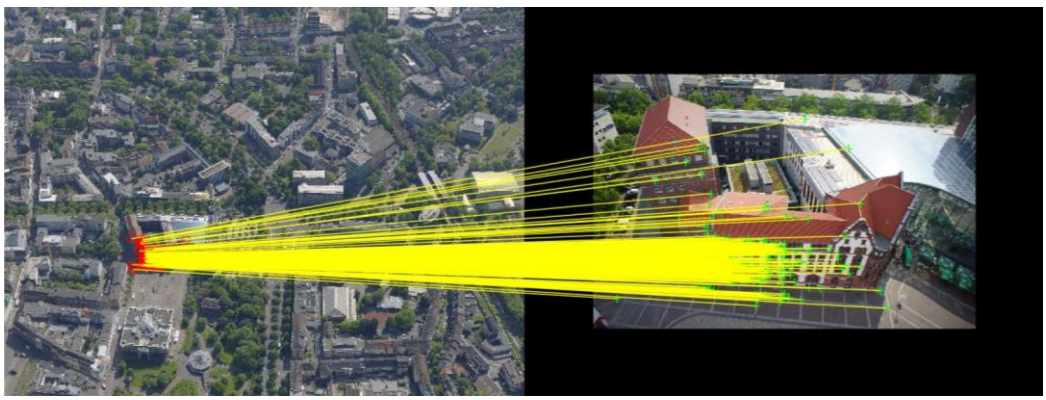


Figure 4.7: Analysis of the number of features detected in the four octaves of the UAV and aerial image after lowering the threshold for feature detection from 0.001 to 0.0001.

Figure 4.8 shows the results achieved after running the algorithm using a threshold of 0.0001 for both images and using features detected in octaves 1 and 2 of the aerial image, and octaves 3 and 4 of the UAV image. 156 correct matches and 38 incorrect matches were observed. The threshold was further lowered to 0.00001. The results improved yet again with a total of 268 putative matches that had 211 correct matches and 57 mismatches.



(a)



(b)

Figure 4.8: (a) Matching results obtained by lowering detection threshold to 0.0001 (b) Matching results obtained by lowering detection threshold to 0.00001.

Main Findings

- Features detected in the lower octaves of lower resolution images, pose as viable candidates for matching with features detected in the higher octaves of higher resolution images.

- Image GSD provides a reliable relationship to automatically select potential octaves that produce features for matching.

- Feature detection is dependent on a threshold. A high threshold value results to less features being detected while a low value results to an increase in the number of features detected.

4.3. Impact of altering feature matching procedures

This step implemented brute force hamming distance, which is meant for matching binary descriptors. knn matching was implemented with k set as 2. This was a necessary step for Lowe's ratio test to be implemented.

It was observed that a significantly high number of putative matches were discarded during the matching step. Two tests were then conducted to ascertain the role played by the Lowe's ratio test and the screening of many-to-1 matches.

Lowe's ratio test was disabled and the pair of descriptors were matched using brute force hamming. The results derived after brute force matching, showed that every point in the left image – aerial image in this case – had a potential match in the right image (UAV image) and this means that many-to-1 matches were present. Recall that the keypoints selected from the aerial image for matching were from octaves 1 and 2. Their total number is lower than the keypoints chosen for matching from the UAV image. Hence the presence of many-to-1 matches which were removed at the end the matching phase. Figure 4.9 shows the matching results achieved and Figure 4.10 shows a sample of a set of many-to-1 matches that were detected.

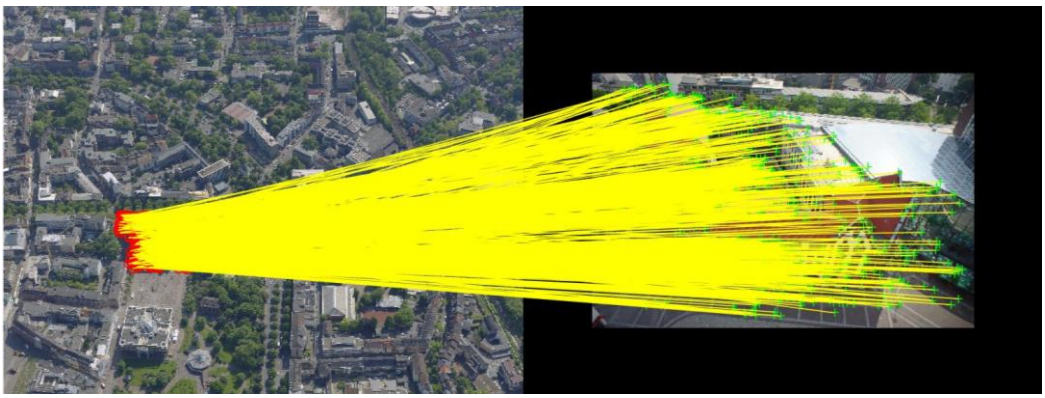


Figure 4.9: Matching results without Lowe's ratio test.

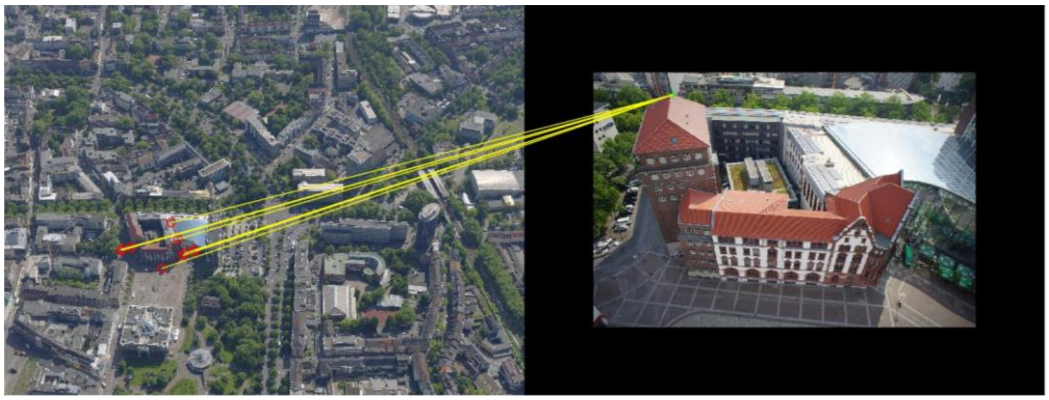


Figure 4.10: A sample of many-to-1 matches.

A visual count was done to quantify the number of correct and incorrect matches within the putative matches illustrated in Figure 4.9. This visual count revealed 630 correct matches out of 1012 putative matches. These figures revealed that the number of correct matches and mismatches increased, proving that Lowe's ratio test was actually discarding a significantly high number of correct matches but also discarding incorrect matches. This can be because features could have been detected very close to each other meaning that their descriptor values are slightly different from each other. So with the ratio test implemented, these matches are discarded and only fewer distinct matches are retained.

Main Findings

- Lowe's ratio test plays a significant role in discarding mismatches but it also plays an equally significant role in discarding correct matches.

- Many-to-1 matches are another source of mismatches.

4.4. Multiple homographies

Although the number of correct matches increased – after disabling the Lowe's ratio test – and they had a better distribution throughout the image, incorrect matches were also present and this could have an effect on the accuracy of the computed F matrix. Another test was conducted to compute matches without outliers. This involved the computation of multiple homographies. Figure 4.11 shows the results achieved after computing multiple homographies with Lowe's ratio test switched off and many-to-1 matches removal was implemented.

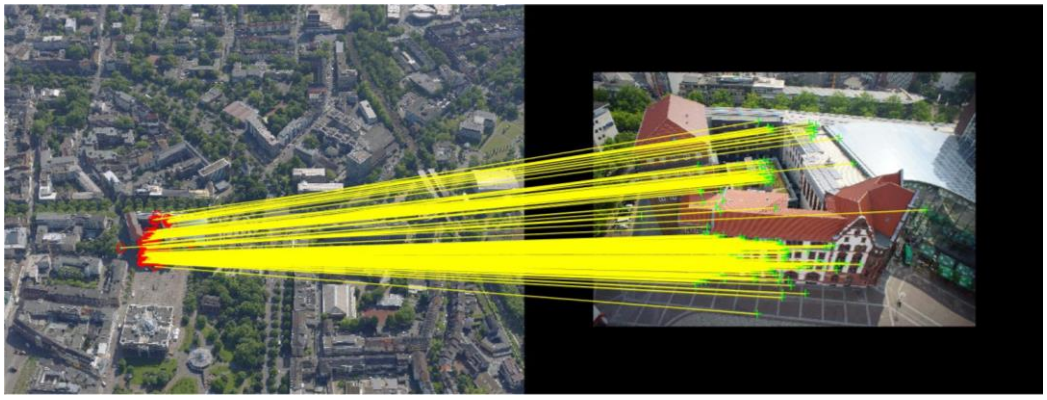


Figure 4.11: Matching results obtained after computing multiple homographies without Lowe's ratio test.

A total of 261 putative matches were computed and it can be seen that the matches are well distributed throughout the overlapping region. Only four outliers were detected.

Another test was conducted to run the multiple homographies with Lowe's ratio test enabled. The iterative computations were set to stop when less than ten points were detected to compute a homography. Figure 4.12 shows the results achieved.

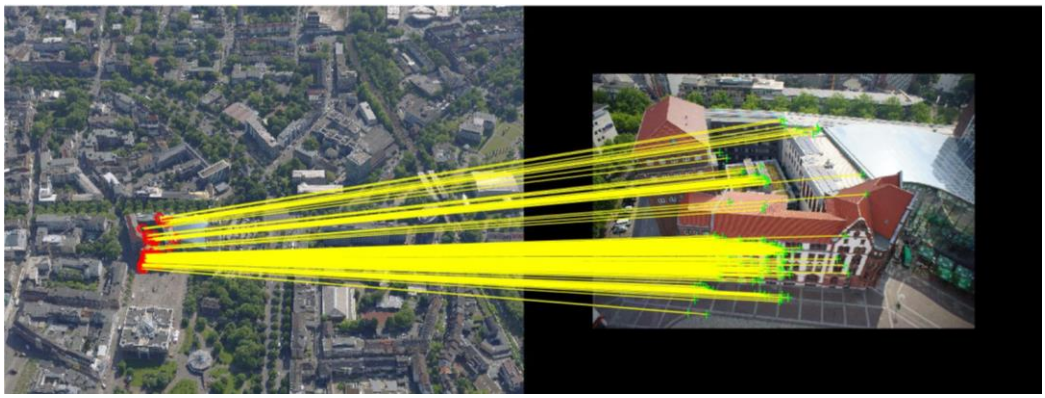


Figure 4.12: Matching results obtained after computing multiple homographies with Lowe's ratio test.

Although less matches were computed (206 in number), all of them were correct without a single outlier. This can be because the ratio test provides a robust solution in retaining distinct matches while computing multiple homographies is reliable in filtering out the few mismatches that evaded the ratio test due to its insensitivity to noise.

Main Findings

- Computing multiple homographies and implementing Lowe's ratio test improves the computation of correct matches and eliminates outliers.

- Computation of multiple homographies provides an alternative to the removal of outliers but this depends on the threshold set – minimum number of points to use – in computing every homography.

4.5. Impact of using Wallis filter

In an attempt to make the algorithm invariant to illumination, Wallis filter was applied as a pre-processing step. Figure 4.13 show the results achieved.

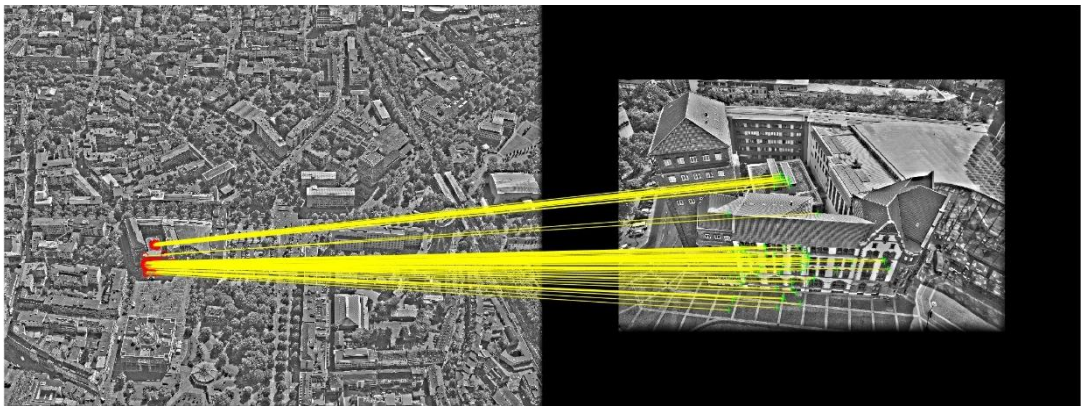


Figure 4.13: Matching done on Wallis filtered images

It was observed that the Wallis filter has up to five parameters (size of the filter, target mean, target standard deviation, contrast and brightness factors) that can be tuned to change the appearance of both images. What was required is that both images look like they all had the same contrast.

Main Findings

- Pre-processing the image pair with the Wallis filter did not improve the results.

4.6. Final algorithm

Table 4.3 shows the parameters used in the final algorithm. In addition, Brute force hamming was used together with knn matching where k was set as 2. Lowe's ratio test was also used and a threshold value of 0.8 was used. Thereafter, multiple homographies were computed with corresponding points not less than ten. The inliers were then used to compute an F matrix using RANSAC with a threshold of 0.001 using the 8 point algorithm.

Table 4.3: Parameters used for feature extraction in the final algorithm

Input	Octaves	Hessian threshold	Descriptor size
UAV image	3, 4	0.00001	64
Aerial image	1, 2	0.00001	64

4.7. Performance evaluation

The final algorithm was then tested on different images that captured different scenes under different conditions such as viewing angle and type of building. Figure 4.14 shows the results achieved when a UAV image, captured at an almost horizontal angle to the building, was matched with an aerial image. 58 correct matches were computed and they were only on the façade of the building.

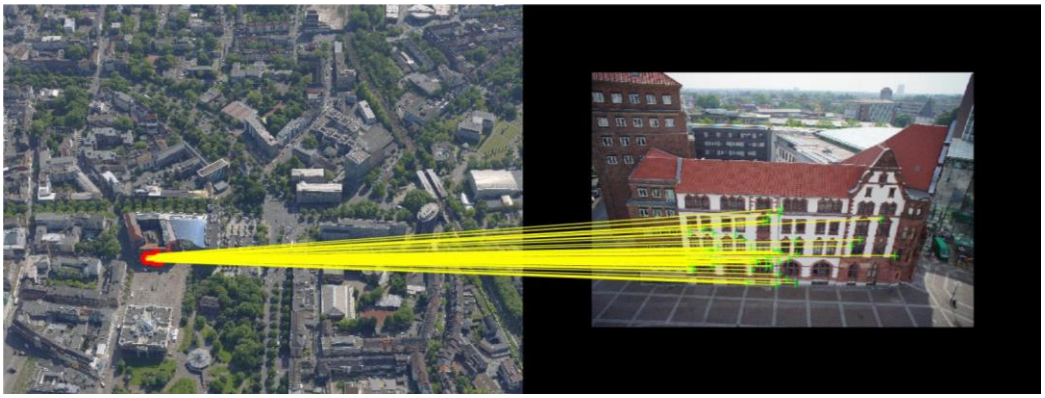


Figure 4.14: 58 correct matches between an aerial image and UAV image with a different viewing angle.

Figure 4.15 shows the results derived from an attempt to match an aerial oblique image and a UAV image that was captured from the side of the building. Unfortunately, no correct matches were computed due to extreme differences in viewing angles between the two images that causes some features to be distorted and occluded in the UAV image and this hampers the process of detecting distinct features in both images.

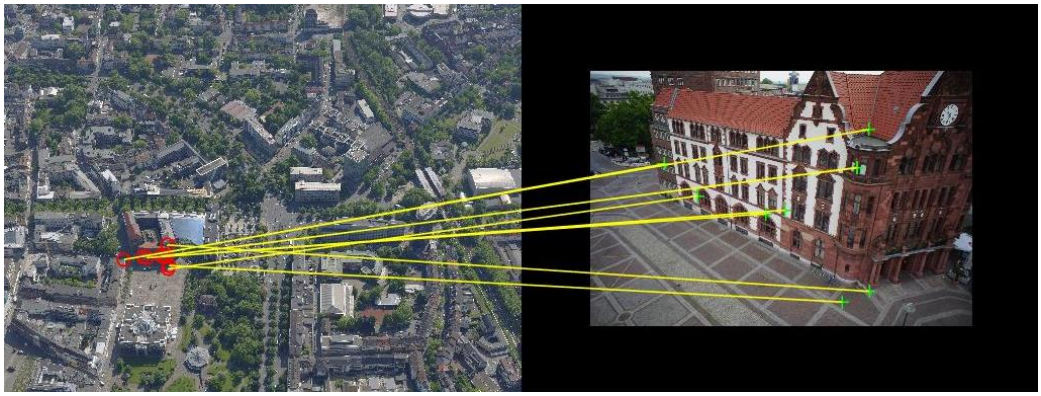


Figure 4.15: Mismatches between an aerial image and a UAV image with different viewing angle.

Figure 4.16 shows the results of matching an aerial image and a UAV image with similar viewing angles for the Rathaus building. 131 correct matches were computed and their distribution was even throughout the scene. The scene captured had a favourable texture that was good for matching.

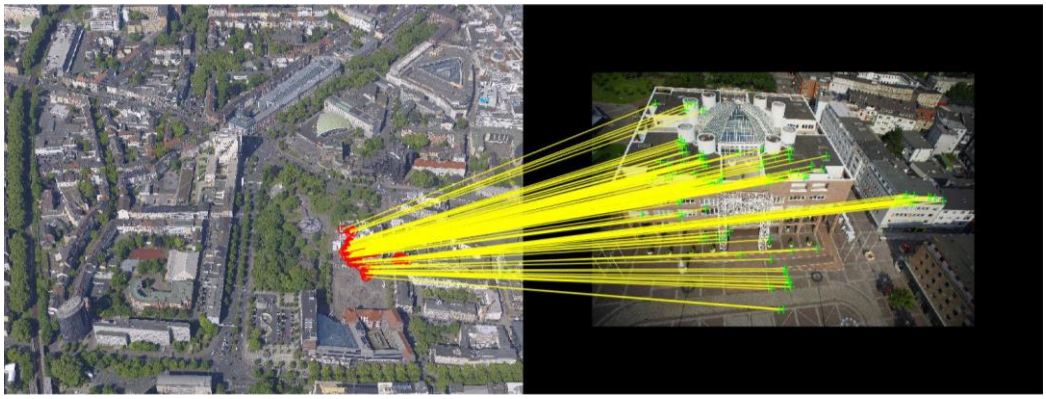


Figure 4.16: 131 correct matches for Rathaus building

Main Findings

- The developed methodology is not invariant to extreme differences in viewing angles between the image acquisition platforms. It is, however, invariant to slight differences in viewing angles.

- The algorithm performed well on Rathaus building scene (as shown in Figure 4.16) however it was still view-dependent.

4.8. Accuracy analysis

Random corresponding points were selected throughout the image pairs and these points were used to compute the F matrix. The results achieved were later compared with the results derived from automatic registration. Figure 4.17 shows the manually selected corresponding points for pair 1 images. The figures for the other pairs are in appendix 3.

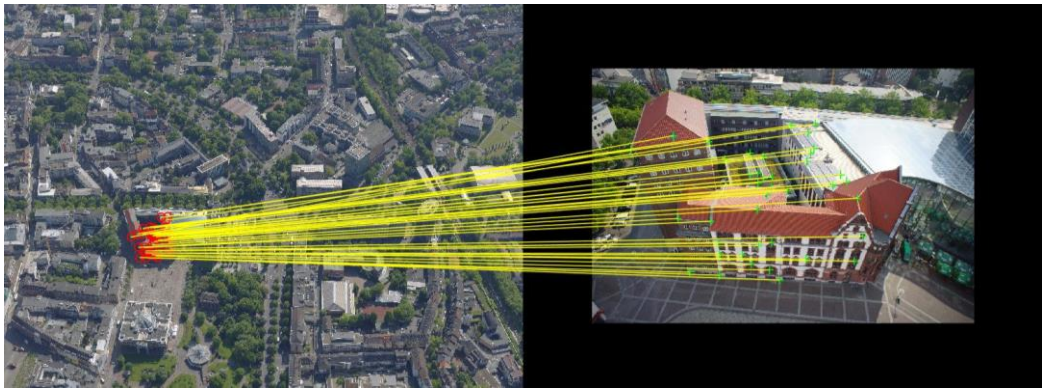


Figure 4.17: Manual registration results

Table 4.4 shows the residual errors computed for all the four case scenarios on the respective UAV images after manual registration. Pair 1 had the least average residual error compared to pair 2, 3 and 4. Followed by pair 2, 4 and 3, in that order. Pair 3 had a higher residual compared to the other pairs possibly due to drastic differences in viewing angles between the cameras during image capture of the pair.

Table 4.4: Residual error results for the different case scenarios after manual registration

Scenario[4]	Number of matches	residual error (pixels)
Pair 1	40	3.12
Pair 2	30	3.26
Pair 3	28	7.75
Pair 4	70	3.67

Table 4.5 shows the residual errors computed for the four scenarios on the respective UAV images after automatic registration. Pair 2 had the least value of residual error but this time followed by pair 1, 4 and once again pair 3 had the largest residual error value. Comparing the residual errors of manual registration and automatic registration, it is clear that manual registration yielded better results than the automatic registration mainly because corresponding features were carefully selected.

Table 4.5: Residual error results for the different case scenarios after automatic registration

Scenario[4]	Number of matches	residual error (pixels)
Pair 1	206	4.91
Pair 2	41	3.45
Pair 3	9	65.86
Pair 4	109	6.34

[4] As illustrated in Figure 3.3

5. DISCUSSION

Surprisingly, AKAZE was able to outperform SIFT, SURF, KAZE, SURF/BRIEF and BRISK by being able to compute more matches than its contenders. This is surprising because, AKAZE employs the use of binary descriptors which has been downplayed for not being reliable due to its nature of being sensitive to noise and hence not as efficient as float descriptors. It is highly suspected that AKAZE was able to outperform SIFT, SURF, KAZE, SURF/BRIEF and BRISK because it retains boundary features which makes it to detect salient features around places like window, roof and wall edges of say buildings. It also uses an FED scheme that is accurate in the localisation of features and it uses a modified version of the Local Binary Descriptors that has proved to result in a high number of matches.

Feature detection is dependent on the characteristics of the scene. It was observed that highly textured surfaces provided stable features in both images whereas features like trees and vegetation provided repetitive features which still cause a challenge to the available descriptors that are not robust enough to uniquely identify these features.

When registering an aerial image and an overlapping UAV image, restricting the search area for matches in the aerial image improves the results of feature matching. This process can be automated by exploiting the exterior orientation of the aerial image and location information of the UAV image encoded in its EXIF tag. However, the developed algorithm has the limitation of not being able to determine the orientation of the UAV image because it is crucial to select an aerial image and a UAV image that are both looking in the same direction so as to discard images that are unlikely to match due to difference in viewing angles.

Scale invariant algorithms detect features in image octaves. The number of features detected in these octaves decays from the lower octaves to the higher octaves. With this understanding in mind, it is crucial to identify which pair of octaves will provide suitable candidates for a successful matching between a pair of images with different resolutions. This was possible by exploiting the GSD at the principal point of the respective images. Since the images were oblique in nature, they have a range of values to represent their GSD. Due to the angle of tilt of the camera, features close to the camera capture more details than features further away. This results to a range of low GSD values to high GSD values from the foreground to the background. This could have resulted to no single combination of octave pairs that provided suitable candidates for matching. Also, feature detection is dependent on the threshold set. AKAZE uses a default threshold of 0.001. Reducing this value led to an increase in the number of detected features and vice versa. Since the UAV image provided suitable candidates for image matching in its higher octaves, the threshold needs to be lowered so as to detect more features in the higher octaves so as to avoid the possibility of detecting few points that were not detected in the aerial image. A similar approach applies to the aerial image. Lowering the threshold makes sure that features are detected throughout the image. In addition, only selecting features that provide good candidates for matching, reduces the problem faced in the matching stage of exhaustively looking for suitable matches because a lot of unnecessary features are discarded at an earlier stage.

When it comes to feature description, AKAZE uses binary values to represent the intensity of neighbouring pixels around the detected feature. Contrary to this, SIFT, SURF and KAZE use float values to represent the same, thus making them more accurate in feature description compared to some binary descriptors. However, AKAZE defied this condition. This may be due to the fact that it employs a different kind of descriptor, M-LDB.

Interesting observations were made during the feature matching tests and outlier removal. The Lowe's ratio test proved to be instrumental in the rejection of outliers. However, it also proved to reject some good

matches. In an attempt to alleviate this drawback, computation of multiple homographies was tested. It was observed that the number of matches increased but they were not all correct. This led to another test that involved using Lowe's ratio test and later computing multiple homographies followed by computation of the fundamental matrix. This led to a result of all correct matches although there wasn't a significant increase in the total number. This observation revealed that the computation of multiple homographies contributes significantly to outlier removal.

The resulting F matrix is supposed to represent a global geometric relationship between the image pair within the areas with matches. In order to have a good representation, then these matches should be well distributed throughout the image so as to have a reliable F matrix. Lack of a good distribution throughout the scene can be because of the characteristics of the scene. For instance, pair 1 detected almost null matches in parts of the images that captured the tiled roofs. Observing these parts of the images closely, it can be observed that the building roof has a uniform contrast making feature detection and feature matching difficult.

Another valid reason why features might have not been matched evenly throughout the image is due to varying illumination. Although the algorithm was able to detect matches despite the images having slight variations in contrast, it was believed that more matches could be computed by pre-processing the images using the Wallis filter. The results were futile maybe because the Wallis filter had five parameters to be tuned and a suitable configuration had to be set to make the pair of images have similar contrast.

On the evaluation of the computed F matrix, the average residual error was computed making use of the average sum of the distances from matched points to their respective epipolar lines. The residual error for the matched points on the UAV image is higher than that for points on the aerial image. This is mainly because of the pixel size and image resolution. The aerial image having a lower resolution will locate features with respect to its pixel size, while the UAV image will do the same but this time its localisation accuracy will be 2-4 times higher than that of the aerial image hence the variations in the residual errors.

In order to define a benchmark that provides a basis for evaluating the accuracy of the F matrix computed by automatic registration, manual registration offers an option. This requires careful identification of corresponding points in both images. If this is done correctly, then when computing the F matrix, RANSAC will detect no outliers.

6. CONCLUSION AND RECOMMENDATIONS

6.1. Conclusion

The main objective of this research was to address the non-trivial problem of multi-resolution image registration between aerial and UAV images. State-of-the-art image matching algorithms were tested and evaluated to determine which algorithm provided promising results for the task at hand. Surprisingly, AKAZE outperforms SIFT, SURF, KAZE, SURF/BRIEF and BRISK, despite using a binary descriptor. A new procedure was then developed to register aerial and UAV images. This procedure implemented the computation of multiple homographies that aided in the rejection of mismatches. Finally, the developed algorithm yielded correct matches that were used to estimate the fundamental matrix between the pair of images. The accuracy of this fundamental matrix was determined comparing its residual error to the one computed from manual registration.

Answers to questions posed at the beginning of this research are given in the following sub section.

6.1.1. Answers to questions

a. What algorithms are available for feature detection/description for the application of registering aerial oblique and UAV images?

Since aerial oblique and UAV images are acquired from different platforms that fly at different heights above the terrain and the on-board cameras have different focal lengths, then the resulting images will have different scales. When it comes to matching these typologies of images, then a scale invariant algorithm will be required for the task.

For an algorithm to be scale invariant, it means that it implements a feature detector that is capable of creating image pyramids and detecting features in these images. So the available feature detectors that are invariant to scale are, for example, SIFT, SURF, BRISK, KAZE and AKAZE, all of which operate differently hence yielding varying results.

b. If these algorithms do exist, what are their drawbacks and can they be modified to make them more reliable in registering multi-resolution images?

SIFT and SURF implement a Gaussian filter that removes boundary features in addition to noise from the images. This can be termed as a drawback because some distinct features are located on window, roof and building edges. Removing these features reduces the chances of identifying correspondences hence the possibility of a low number of matches. Moreover, SIFT is expensive to compute compared to SURF, KAZE and the binary descriptors.

Binary descriptors have been reported to be inefficient compared to float descriptors when it comes to matching features accurately. The results of the research proved that this might not hold for all binary descriptors but the claim might hold for say, BRIEF and BRISK which didn't yield satisfactory results.

Lastly, all of the algorithms demonstrated the ability of being able to be improved upon. This is so because they all had adjustable parameters that the user can tune depending on the application at hand. In order for one to make these algorithms reliable for a particular application, one needs to figure out the effects played by each of these adjustable parameters.

c. *What strategies can be utilised to develop an algorithm for the registration of multi-scale (scale range of between 2-4 times) images?*

The following strategies can be utilised:

- Selection of a suitable feature detector/descriptor that is invariant to scale changes.
- Detection of similar features throughout the image pair in all octaves. This is possible by adjusting the threshold for feature detection. A suitable threshold ought to be set to ensure a sufficient amount of features are detected in both images and in the octaves that are likely to produce matching features i.e. the lower octaves of the lower resolution image and the higher octaves of the higher resolution image.
- If EO parameters of the low resolution image are present and the high resolution image has geolocation information embedded in its EXIF tag, then a restricted area can be defined around the low resolution image on the area of overlap between the pair of images to be registered.
- The use of multiple homographies can be used to reduce the number of outliers making the process of computing the F matrix to give out a reliable result i.e. an F matrix will be computed with possibly all correct.

d. *Which step of image registration plays a crucial role in the registration process of multi-resolution images?*

This depends on the application at hand. In the case of this research it was noticed that the stage of feature detection played a crucial in the whole registration process. Since features were found to be matching between low octaves of the aerial image and high octaves of the UAV image, and the number of detected features decays from low to high octaves, then with this understanding, it was mandatory to adjust the detection threshold so as to detect more features in the higher octaves of the UAV image. This being the initial stage of the process, distinct features need to be detected in both images so that the subsequent stages can have reliable inputs that will lead to a successful registration.

e. *What influence does GNSS/IMU information have on multi-scale image registration?*

Presence of GNSS/IMU information proved to be crucial in restricting the search area for matches in the image covering a large area. By doing this, then the efficiency of locating correct matches is increased.

In this research, this was made possible by implementing the collinearity equations to first determine the ground footprint of the aerial image and then look for the UAV image that's within this footprint and back project its position to the aerial image. This was possible with the assumption that the UAV image was taken by a camera with an almost similar viewing angle as the aerial image since orientation parameters for UAV camera were not availed.

Moreover, the information can later be used to estimate coarse orientation parameters of the UAV image which has only GPS information availed.

f. *How reliable is the developed algorithm?*

The developed algorithm performs well for multi-resolution images that are taken with almost similar viewing directions. When the viewing angle changes drastically then the performance is hampered. In

addition, the algorithm is dependent on the type of scene that was captured. The images used for this research captured building scenes. Favourable results were derived from buildings with good texture and varying contrast. It is because of this reason that a huge number of matches were not detected all over the overlapping area of the images.

The computed residual errors also demonstrated the reliability of the computed F matrix. Image pairs that produced a high number of correct matches had small averages of the residual errors.

6.2. Recommendations

The next stage would be to determine the relative orientation of the registered UAV image.

Further research that can be looked into is the registration of terrestrial images to UAV image and possibly aerial images. The orientation of terrestrial images are also unknown and their registration to already oriented UAV images can reveal crucial information that can be exploited in the generation of multi-resolution 3D scenes.

Additionally, other feature matching techniques, like graph matching, can be tested so as to try and improve the total number of features matched throughout the image pairs.

LIST OF REFERENCES

Agarwal, S., Furukawa, Y., Snavely, N., Curless, B., Seitz, S. M., & Szeliski, R. (2010). Reconstructing Rome. *Computer*, *43*(6), 40–47. http://doi.org/10.1109/MC.2010.175

Alcantarilla, P. F., Bartoli, A., & Davison, A. J. (2012). KAZE features. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *7577 LNCS*(PART 6), 214–227. http://doi.org/10.1007/978-3-642-33783-3_16

Alcantarilla, P. F., Nuevo, J., & Bartoli, A. (2013). Fast explicit diffusion for accelerated features in nonlinear scale spaces. *British Machine Vision Conference*, 13.1-13.11. http://doi.org/10.5244/C.27.13

Bay, H., Tuytelaars, T., & Van Gool, L. (2006). SURF: Speeded up robust features. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *3951 LNCS*, 404–417. http://doi.org/10.1007/11744023_32

Calonder, M., Lepetit, V., Strecha, C., & Fua, P. (2010). BRIEF : Binary robust independent elementary features. *European Conference on Computer Vision (ECCV)*, 778–792. http://doi.org/10.1007/978-3-642-15561-1_56

Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *8*(6), 679–698. http://doi.org/10.1109/TPAMI.1986.4767851

Chen, M., Zhu, Q., Huang, S., Hu, H., & Wang, J. (2016). Robust low-altitude image matching based on local region constraint and feature similarity confidence, *III*(July), 12–19. http://doi.org/10.5194/isprsannals-III-3-19-2016

Figat, J., Kornuta, T., & Kasprzak, W. (2014). Performance evaluation of binary descriptors of local features. *Computer Vision and Graphics*, *8671*, 187–194. http://doi.org/10.1007/1-4020-4179-9

Fischler, M. a, & Bolles, R. C. (1981). Random sample consensus: A paradigm for model fitting with applicatlons to image analysis and automated cartography. *Communications of the ACM*, *24*(6), 381–395. http://doi.org/10.1145/358669.358692

Förstner, W., & Gülch, E. (1987). A Fast operator for detection and precise location of distinct points, corners and centres of circular features. *Proceedings of ISPRS Intercommission Conference on Fast Processing of Photogrammetric Data*.

Frahm, J.-M., Georgel, P. F., Gallup, D., Johnson, T., Raguram, R., Wu, C., … Lazebnik, S. (2010). Building Rome on a cloudless day. *Proceedings of the European Conference on Computer Vision*, 368–381.

Fraundorfer, F., & Bischof, H. (2005). A novel performance evaluation method of local detectors on non-planar scenes. *Computer Vision and Pattern Recognition - Workshops, 2005*, 33. http://doi.org/10.1109/CVPR.2005.393

Gautama, S., Goeman, W., & D'Haeyer, J. (2004). Robust detection of road junctions in VHR images using an improved ridge detector. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *XXXV*, *Part*, 1682–1777.

Geniviva, A., Faulring, J., & Salvaggio, C. (2014). Automatic georeferencing of imagery from high-resolution, low-altitude, low-cost aerial platforms. *Geospatial InfoFusion and Video Analytics IV; and Motion Imagery for ISR and Situational Awareness II*, *9089*. http://doi.org/10.1117/12.2050493

Gerke, M., Nex, F., & Jende, P. (2016). Co-registration of terrestrial and UAV-based images – Experimental results, *XL*(February), 10–18. http://doi.org/10.5194/isprsarchives-XL-3-W4-11-2016

Gonzales, R. C., Woods, R. E., & Eddins, S. L. (2014). *Digital image proccesing using MATLAB*. Pearson Prentice Hall. http://doi.org/10.1007/s13398-014-0173-7.2

Goshtasby, A. A. (2012). *Image registration: Principles, tools and methods*. *Advances in Computer Vision and Pattern Recognition*. http://doi.org/10.1007/978-1-4471-2458-0_1

Grewenig, S., Weickert, J., & Bruhn, A. (2010). From box filtering to fast explicit diffusion. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *6376 LNCS*, 533–542. http://doi.org/10.1007/978-3-642-15986-2_54

Harris, C., & Stephens, M. (1988). A combined corner and edge detector. *Procedings of the Alvey Vision Conference 1988*, 147–151. http://doi.org/10.5244/C.2.23

Hartley, R., & Zisserman, A. (2004). *Multiple view geometry in computer vision* (2nd ed.). Cambridge University Press.

Hartmann, W., Havlena, M., & Schindler, K. (2015). Recent developments in large-scale tie-point matching. *ISPRS Journal of Photogrammetry and Remote Sensing*, *115*, 47–62. http://doi.org/10.1016/j.isprsjprs.2015.09.005

Heinly, J., Schönberger, J. L., Dunn, E., & Frahm, J. M. (2015). Reconstructing the world* in six days *(as captured by yahoo 100 million image dataset). *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 3287–3295. http://doi.org/10.1109/CVPR.2015.7298949

Jazayeri, I., & Fraser, C. S. (2008). Interest operators in close-range object reconstruction. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, Beijing*, *37*(B5), 69–74. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.155.1585&rep=rep1&type=pdf

Jende, P., Peter, M., Gerke, M., & Vosselman, G. (2016). Advanced tie feature matching for the registration of mobile mapping imaging data and aerial imagery. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *XLI*(July), 617–623. http://doi.org/10.5194/isprsarchives-XLI-B1-617-2016

Jiang, G., Liu, L., Zhu, W., Yin, S., & Wei, S. (2015). A 181 GOPS AKAZE accelerator employing discrete-time cellular neural networks for real-time feature extraction. *Sensors*, *15*(9), 22509–22529. http://doi.org/10.3390/s150922509

Koch, T., Zhuo, X., Reinartz, P., & Fraundorfer, F. (2016). A new paradigm for matching UAV- and aerial images. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, *III*(July), 12–19. http://doi.org/10.5194/isprsannals-III-3-83-2016

Krig, S. (2014). Interest point detector and feature descriptor survey. *Computer Vision Metrics*, (1), 217–282. http://doi.org/10.1007/978-1-4302-5930-5

Lee, P., & Timmaraju, A. S. (2014). Learning binary descriptors from images, 1–5. Retrieved from http://cvgl.stanford.edu/teaching/cs231a_winter1415/prev/projects/LeeTimmaraju_report.pdf

Leutenegger, S., Chli, M., & Siegwart, R. Y. (2011). BRISK: Binary robust invariant scalable keypoints. *Proceedings of the IEEE International Conference on Computer Vision*, 2548–2555. http://doi.org/10.1109/ICCV.2011.6126542

Levi, G., & Hassner, T. (2015). LATCH: Learned arrangements of three patch codes, *arXiv 1501*.

Longuet-Higgins, H. C. (1981). A computer algorithm for reconstructing a scene from two projections. *Nature*, *293*, 133–135.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. *Proceedings of the Seventh IEEE International Conference on Computer Vision*, *2*(8), 1150–1157. http://doi.org/10.1109/ICCV.1999.790410

Lowe, D. G. (2004). Distinctive image features from scale invariant keypoints. *International Journal of Computer Vision*, *60*, 91–110. http://doi.org/http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94

Luong, Q.-T., & Faugeras, O. D. (1997). The fundamental matrix: Theory, algorithms, and stability analysis. *International Journal of Computer Vision*, *17*(1996), 43–75.

Mathworks. (2012). Estimate fundamental matrix from corresponding points in stereo images - MATLAB estimateFundamentalMatrix - MathWorks Benelux. Retrieved February 2, 2017, from https://nl.mathworks.com/help/vision/ref/estimatefundamentalmatrix.html

Mikolajczyk, K., & Schmid, C. (2005). A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*(10), 1615–1630. http://doi.org/10.1109/TPAMI.2005.188

Miksik, O., & Mikolajczyk, K. (2012). Evaluation of local detectors and descriptors for fast feature matching. *International Conference on Pattern Recognition*, 2681–2684. http://doi.org/978-1-4673-2216-4

Morel, J.-M., & Yu, G. (2009). ASIFT: A new framework for fully affine invariant image comparison. *SIAM Journal on Imaging Sciences*, *2*(2), 438–469. http://doi.org/10.1137/080732730

Nex, F., Gerke, M., Remondino, F., Przybilla, H.-J., Bäumker, M., & Zurhorst, A. (2015). ISPRS benchmark for multi-platform photogrammetry. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, *II-3/W4*(March), 135–142. http://doi.org/10.5194/isprsannals-II-3-W4-135-2015

Nex, F., & Jende, P. (2016). Feature detection, feature description, feature matching 2/2 [Powerpoint slides]. Retrieved from https://blackboard.utwente.nl/bbcswebdav/pid-958999-dt-content-rid-2138907_2/xid-2138907_2

Nex, F., & Remondino, F. (2014). UAV for 3d mapping applications: A review. *Applied Geomatics*, *6*(1), 1–15. http://doi.org/10.1007/s12518-013-0120-x

OpenCV. (2012). OpenCV: AKAZE class reference. Retrieved February 12, 2017, from http://docs.opencv.org/trunk/d8/d30/classcv_1_1AKAZE.html

Pieropan, A., Björkman, M., Bergström, N., & Kragic, D. (2016). Feature descriptors for tracking by detection: A benchmark. Retrieved from http://arxiv.org/abs/1607.06178

Prewitt, J. M. S. (1970). Object enhancement and extraction. *Picture Processing and Psychopictorics*, 75–149.

Roberts, L. G. (1963). *Machine perception of three-dimensional solids* (Doctral dissertation, Massachusetts Institute of Technology).

Rosten, E., & Drummond, T. (2006). Machine learning for high-speed corner detection. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *3951 LNCS*, 430–443. http://doi.org/10.1007/11744023_34

Shan, Q., Adams, R., Curless, B., Furukawa, Y., & Seitz, S. M. (2013). The visual turing test for scene

reconstruction. *Proceedings - 2013 International Conference on 3D Vision, 3DV 2013*, 25–32. http://doi.org/10.1109/3DV.2013.12

Sobel, I. (1990). An isotropic 3 by 3 image gradient operator. *Machine Vision for Three-Dimensional Sciences*, *1*(1), 23–34. Retrieved from http://ci.nii.ac.jp/naid/10018992790/

Szpak, Z. L., Chojnacki, W., Eriksson, A., & Van Den Hengel, A. (2014). Sampson distance based joint estimation of multiple homographies with uncalibrated cameras. *Computer Vision and Image Understanding*, *125*, 200–213. http://doi.org/10.1016/j.cviu.2014.04.008

Trzcinski, T., & Lepetit, V. (2012). Efficient discriminative projections for compact binary descriptors. *European Conference on Computer Vision (ECCV)*, *7572*, 228–242. http://doi.org/10.1007/978-3-642-33718-5_17

Tuytelaars, T., & Mikolajczyk, K. (2008). Local invariant feature detectors: A survey. *Computer Graphics and Vision*, *3*(3), 177–280. http://doi.org/10.1561/0600000017

Ünsalan, C., & Sirmacek, B. (2012). Road network detection using probabilistic and graph theoretical methods. *IEEE Transactions on Geoscience and Remote Sensing*, *50*(11), 4441–4453.

Wallis, K. F. (1979). Seasonal adjustment and relations between variables (pp. 347–364).

Yang, T.-Y., Lin, Y.-Y., & Chuang, Y.-Y. (2016). Accumulated stability voting: A robust descriptor from descriptors of multiple scales. http://doi.org/10.1109/CVPR.2016.42

Yang, X., & Cheng, K. T. (2012). LDB: An ultra-fast feature for scalable augmented reality on mobile devices. *ISMAR 2012 - 11th IEEE International Symposium on Mixed and Augmented Reality 2012, Science and Technology Papers*, *2*, 49–57. http://doi.org/10.1109/ISMAR.2012.6402537

Zitová, B., & Flusser, J. (2003). Image registration methods: A survey. *Image and Vision Computing*, *21*(11), 977–1000. http://doi.org/10.1016/S0262-8856(03)00137-9

Zuliani, M., Kenney, C. S., & Manjunath, B. S. (2005). The multiransac algorithm and its application to detect planar homographies. *Proceedings - International Conference on Image Processing, ICIP*, *3*, 153–156. http://doi.org/10.1109/ICIP.2005.1530351

APPENDICES

Appendix 1: Results from SIFT, SURF, KAZE, SURF/BRIEF and BRISK for an uncropped aerial image.

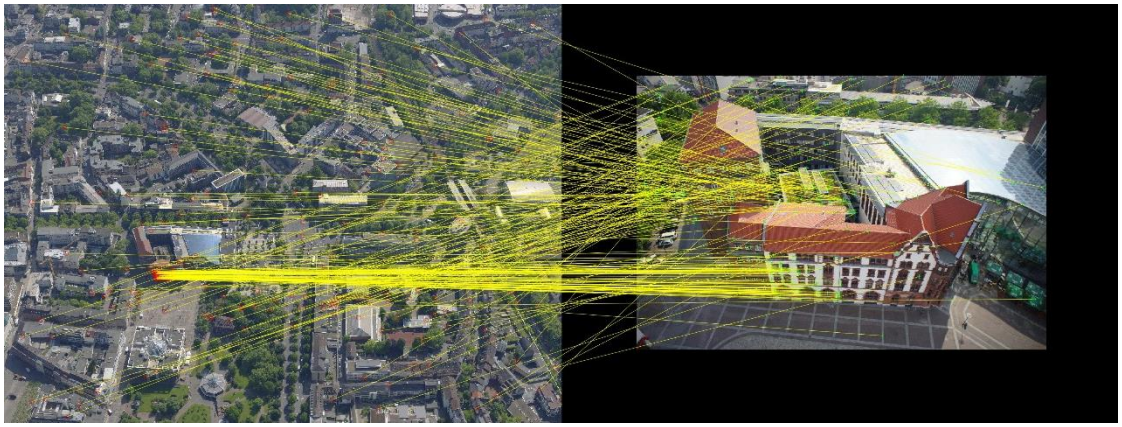


Figure A 1: SIFT

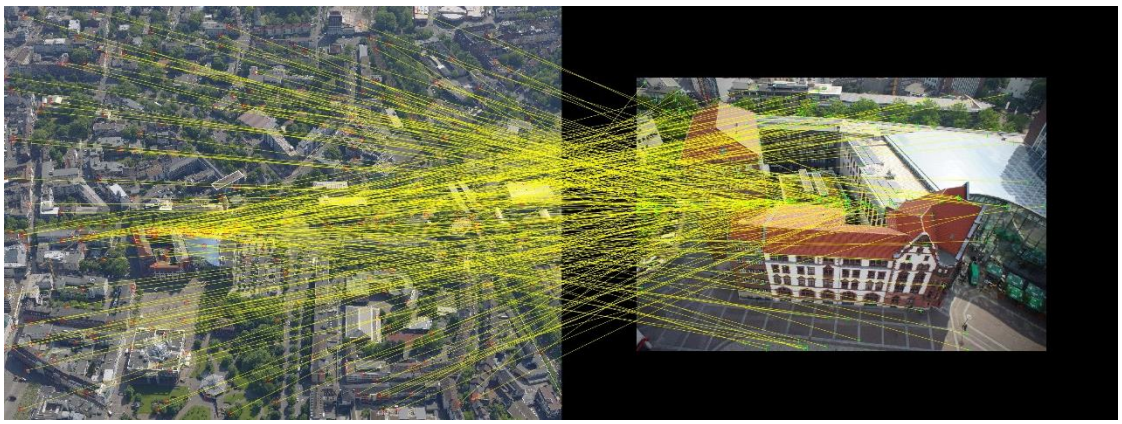


Figure A 2: SURF

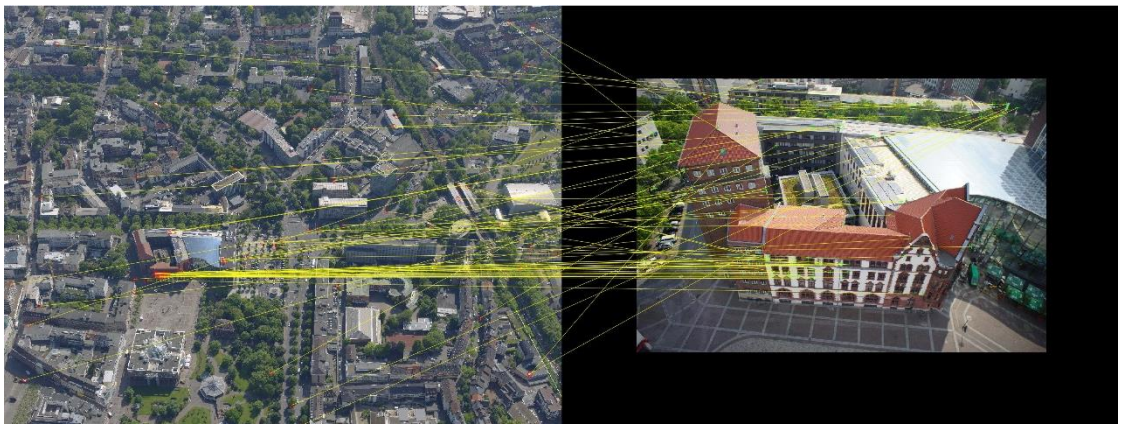


Figure A 3: KAZE

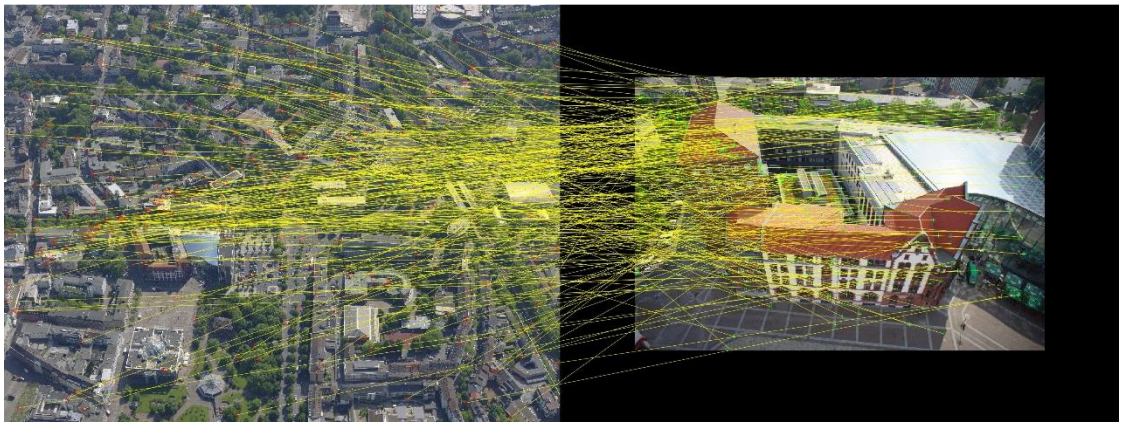


Figure A 5: SURF/BRIEF

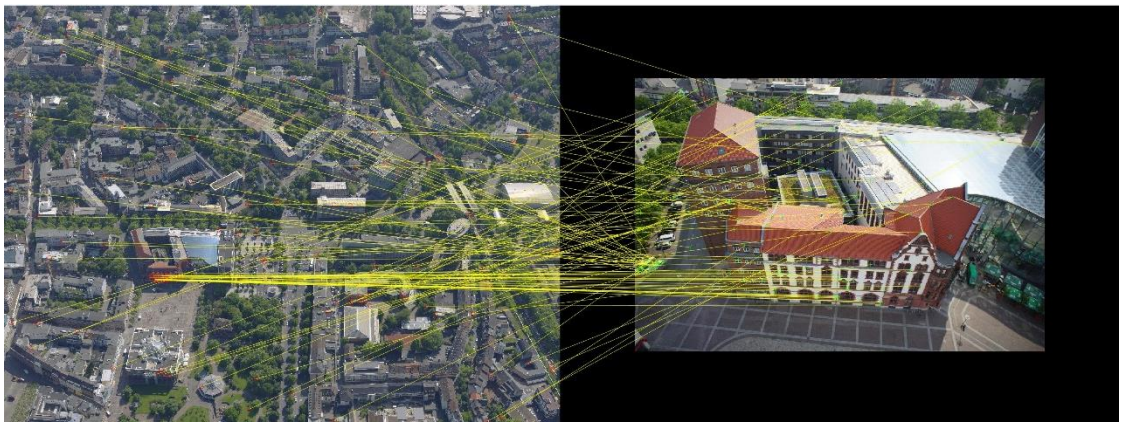


Figure A 4: BRISK

Appendix 2: Results from SIFT, SURF, KAZE, SURF/BRIEF and BRISK for a cropped aerial image.

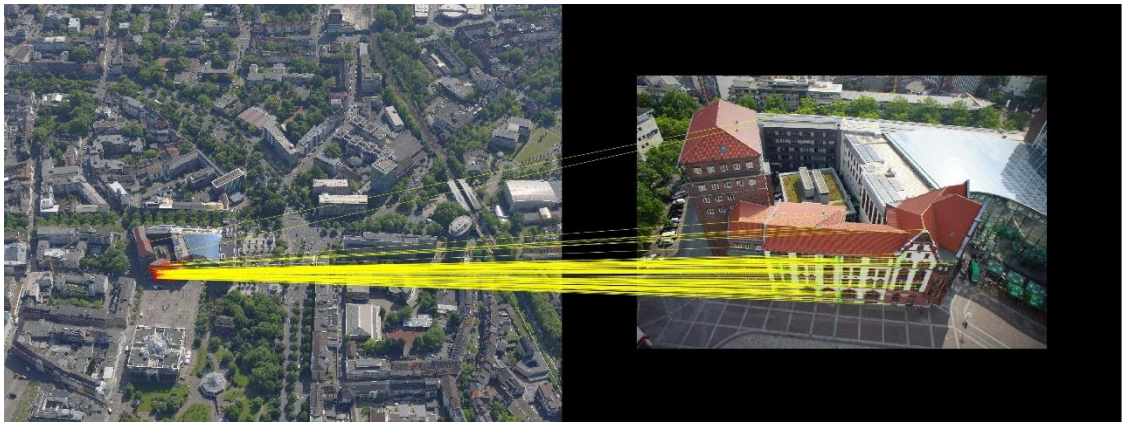


Figure A 6: SIFT

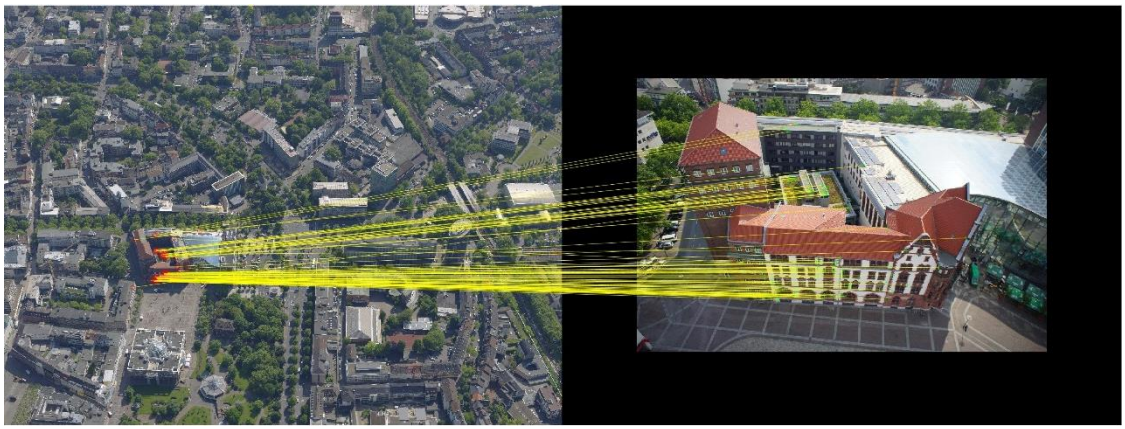


Figure A 7: SURF

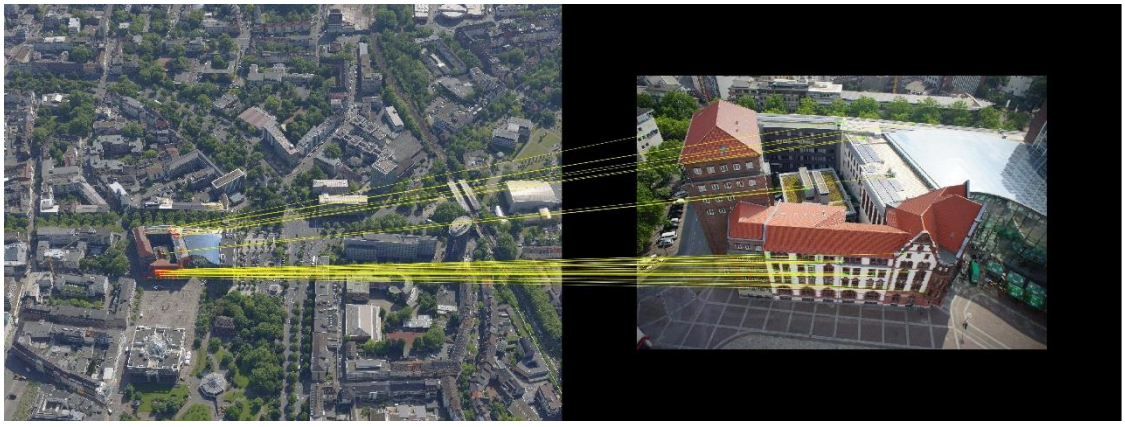


Figure A 8: KAZE

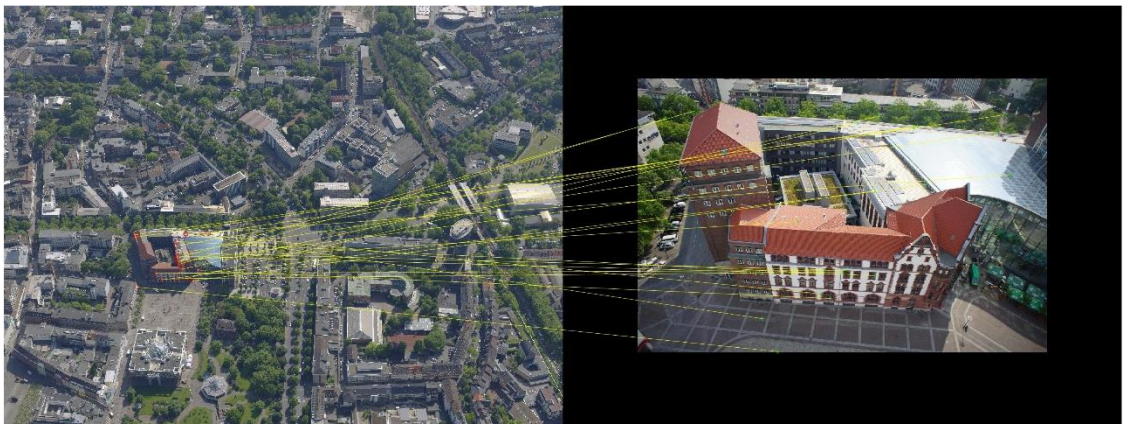


Figure A 9: SURF/BRIEF

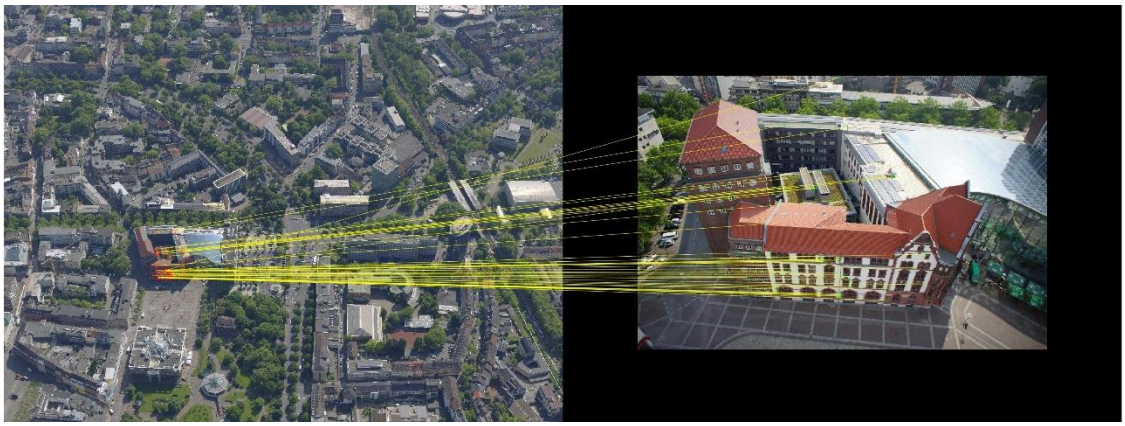


Figure A 10: BRISK

Appendix 3: Results from manual registration

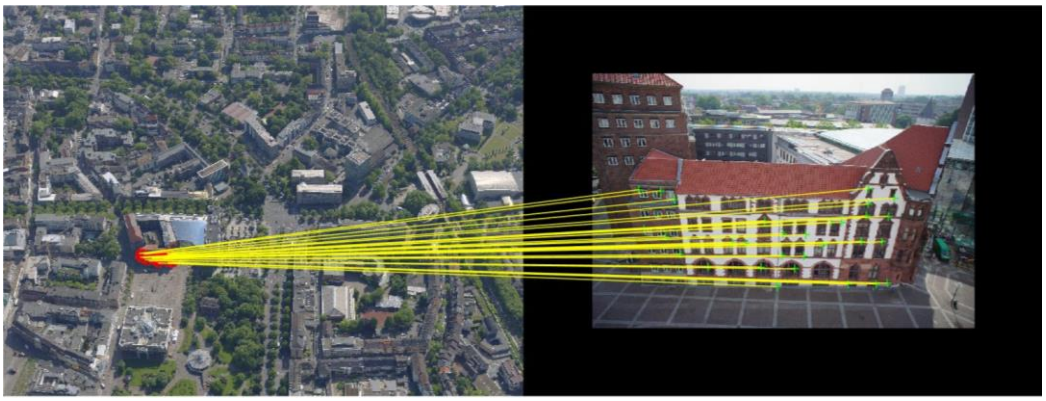


Figure A 11: Pair 2

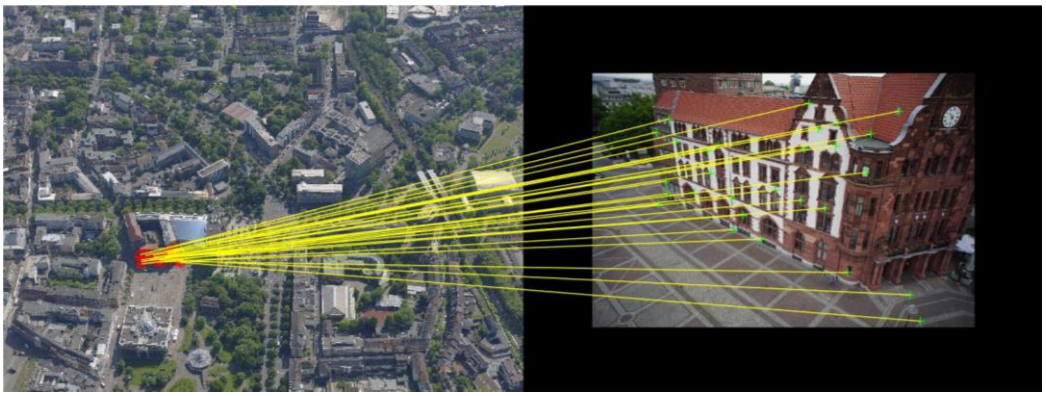


Figure A 12: Pair 3

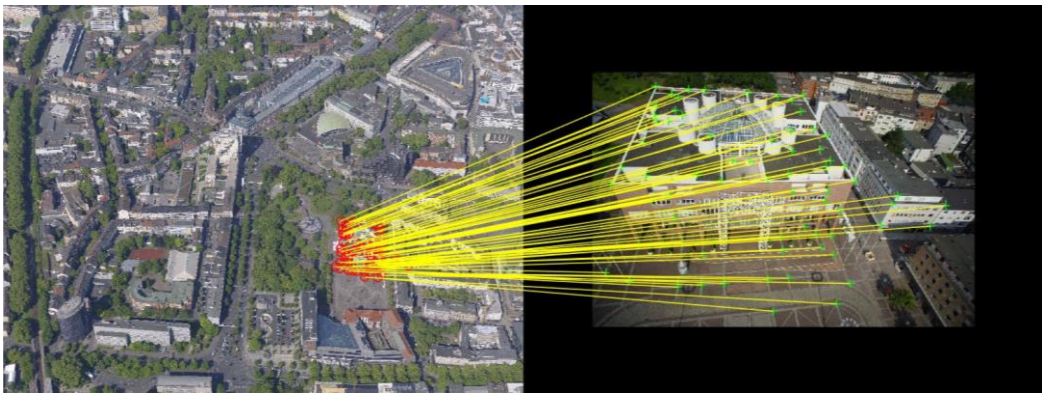


Figure A 13: Pair 4