CRANFIELD UNIVERSITY

Alexander William Elliott

VISION BASED LANDMARK DETECTION FOR UAV NAVIGATION

SCHOOL OF ENGINEERING

Supervisor: Al Savvaris

MRes THESIS

Abstract

The majority of Unmanned Aerial Vehicles (UAV) available today depend on Global Position Satellites (GPS) and inertial measurement units (IMU) for state estimation used in navigation and control. However with the increase in availability of cheap GPS jamming technologies leads to concerns over the dependence of GPS for control and navigation. A possible solution is to use a downward looking camera on-board the aircraft, and using vision based techniques the aircraft can estimate its position without the need for GPS signals.

The focus of this thesis is to develop reliable methods for feature and landmark extraction for use with the vision based positioning system.

The first method proposed estimated the aircraft position in real time using Image Registration techniques, during testing it was found that it did not cope well if there are differences between the source and reference images, which could be due to seasonal or lighting changes. To overcome this problem, work was conducted to look at object detection (buildings, and roads) which enable objects to be detected despite changes in season, or lighting conditions. Three such methods are presented in this thesis, although all of them have been shown to work, only the Haar classifier based method is suitable for use on-board a UAV as the other methods are computationally intensive. Further testing of the Haar classifier was conducted to investigate the full envelope of the object detection under a simulated test.

Haar classifier cascade for object detection in aerial images was shown to be capable of detecting objects reliably under a variety of different situations in this thesis. This information can then be used with a GIS database to match the objects extracted from the image, with objects on a geo coded object database to estimate the aircraft position in a variety of different conditions.

Acknowledgements

I would like to thank Dr A Savvaris for providing the opportunity and support to complete my MSc by Research Thesis

I would also like to thank all those who have helped and supported me during this thesis.

Contents

1 Introduction

The majority of Unmanned Aerial Vehicles (UAV) available today depend on Global Position Satellites (GPS) and inertial measurement units (IMU) for state estimation used in navigation and control. The GPS signal is used to correct the drift rate in the IMU. Typically, smaller UAVs use lower performance IMUs due to limited payload capacity and cost. These IMUs rely heavily on GPS for drift compensation and navigation. Such sensors are use on most small UAVs; however, the loss of the GPS system for extended periods of time can lead to system failure. The GPS signal can become unreliable due to interference, and signal multipath reflections when operating close to obstacles. Furthermore, the availability of low cost GPS jamming technology on the market has led to concerns about the dependence of the GPS for UAV control and navigation.

Project Athena, a research program managed by Cranfield University is looking at integrated technologies for a Medial Altitude Long Endurance UAV. This project is investigating many novel technologies for use onboard the Aegis UAV. These technologies include a fuel cell to generate power for an electric engine, satellite communications, computer vision algorithms for 3D mapping, target tracking, and a vision based positioning system.

This thesis investigates alternative methods for UAV navigation using computer vision with a focus on the image analysis aspects that will feed into the vision based positioning system. The vision based system will replace the GPS signal allowing the UAV to know its position using geo referenced aerial imagery. This is possible with the easy and free access to satellite and aerial imagery of the world (Google Earth and Virtual Earth). This system will allow for safer navigation of UAVs that are resilient to GPS jamming and outages.

1.1 Objectives

The main aim of this project is to develop a robust system for detecting landmarks in aerial images. The main objectives of the project are described below:

- **Review methods and techniques currently used in visual navigation on UAV's**
  It is important to conduct a literature review related to the topic of the project to establish what the state of the art currently is in the related research areas, and also to identify gaps and future work. This will also minimise the risk of reinventing the wheel

- **Identify suitable landmarks for a UAV vision based positioning system**
  In an aerial image, there are many landmarks (salient features) and it is important to distinguish which ones can reliably be found, and which are suitable. static features such as buildings, roads are important to be distinguished from moving features such as cars.

- **Develop/implement methods for robust landmark detection in aerial images**
  From the literature review the best methods will be identified, these methods will then be implemented and modified to fit the application of a geolocation system. As the system will run on-board the UAV, it is important to make the implementation as robust as possible.

- **Validate algorithms/methods for landmark detection and classification**
  Once the methods have been developed it is very important to validate and benchmark them to see how well they perform in varying real world situations as the ultimate goal is to run this system on-board a UAV with a downward looking camera.

1.2 Thesis Layout

The first chapter of this thesis gives a brief introduction of the work to be carried out in this thesis as well as the aims and objectives.

The second chapter covers some basic information around this thesis, including an overview of the vision based positioning system, and a literature review of vision based positioning, and also general object detection from UAVs. This will establish the current state of the art in this research area and help give guide of the work that needs to be covered in this thesis to make a contribution to this area.

The third chapter looks at using image registration techniques for visual based positioning using the SURF key point detector. This chapter goes of the details of the implementation used in this thesis, as well as testing the suitability of such a system for visual based positioning onboard an aircraft.

Chapters 4 and 5 look at using edges and colour information to detect road intersections and buildings. The motivation behind detecting objects for visual based positioning systems are also described in these chapters. These methods were shown to work, however the main disadvantage was the long processing time, particularly for the road detection method.

Chapter 6 looks at using Haar classifiers for object detection. Haar classifiers were considered as they are much faster computationally. This chapter covers the details of the Haar classifier implementation used in this thesis, as well as the details on how the classifier was trained using matching learning techniques. In Chapter 7a direct comparison was conducted on the same dataset for the different methods of landmark detection.

Once a variety of different methods have been tested in the previous chapters, chapter 8 takes the most promising method (Haar classifier) and looks at a case study to fully test the suitability of this method for use with a vision based positioning system.

Finally the Chapter 9 is an evaluation and conclusion the work conducted in this theses. This chapter also gives details on future work and improvements.

2 Background Information

2.1 Vision Based Positioning System

A basic overview of how the vision based positioning system works is shown below in Figure 2-1. This project focuses on the area within the dashed box (Landmark Extraction)

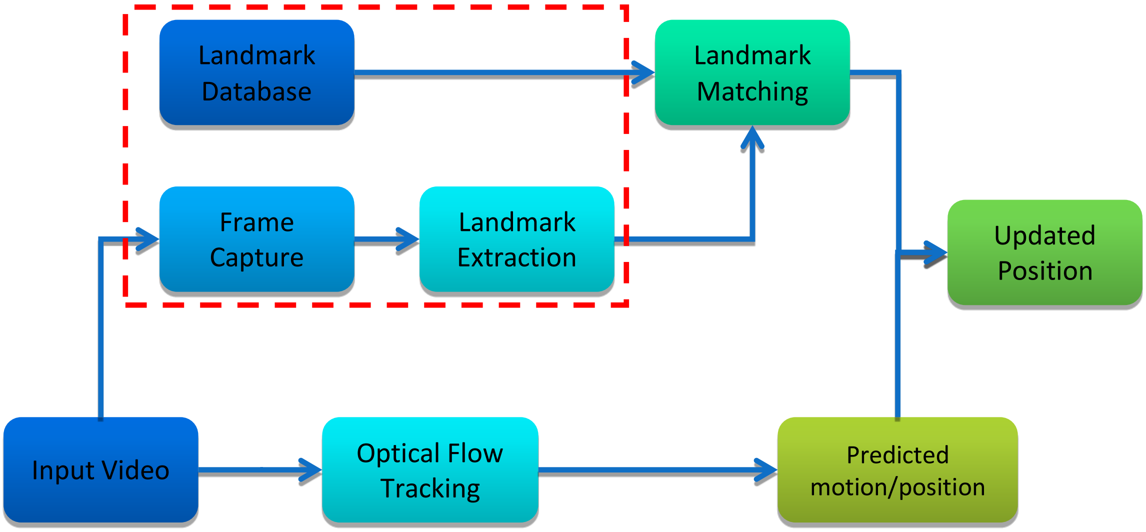


Figure 2-1 Vision based positioning system diagram

The vision based positioning system contains a database of feature points of an area, were each feature point can be a representation of a fixed object in the real world such as a building, bridge, road, etc. The relationships between each of the points are recorded and stored on the database as well as their corresponding world location. The main key relationship is the angle between each of the points. This database can be obtained by extracting points from an existing GIS system or by analysis of a database of aerial images such as Google earth. The database will be prepared offline and stored on board the UAV. During a mission an on board downward looking camera captures images in real time of the terrain the UAV is flying over. This image is processed in order to detect and extract useful features such as buildings, roads etc. The feature extraction is the main focus of this thesis. These detected features are then sent to the geolocation system which will match the detected features with the database of features in order to establish the current location of the UAV. The process of geolocation and feature detection is computationally expensive and is not possible to run in real time, therefore optical flow algorithms will be running in order to provide intermediate positioning between the geolocation updates. The optical flow tracking is able to run in real time, however the estimated position drifts with time due to noise, so the optical flow system is used in between the updates from landmark matching. Once the landmarks have been matched the optical flow position estimate is combined with the new position estimate to output an accurate position of the UAV,.

2.2 Related Work

This section investigates related research in the field of vision based positioning in the form of a literature review. This will give an overview on how similar problems are solved which will give a good understanding and basis for work conducted in this thesis.

There are two main methods that can be used for vision based geolocation, Image Registration, and Landmark Detection:

- ***Image Registration***
  Image Registration techniques can match and align two images onto one another. For Vision based geolocation, a source image taken onboard the aircraft/UAV is then match and aligned onto a database of reference images. One the image has been matched, a position can be estimated as the real world position of the reference database images are known. As discussed later this method has been proven to work, but does not perform well when there are differences between the source and reference images due to seasonal or lighting changes. Image registration techniques are also traditionally computationally expensive as they compare and match pixel intensities of the source and reference images

- ***Landmark Based***
  This approach attempts to extract useful landmarks (buildings, roads, rivers) from an aerial image. This method is overcomes the need for up to date reference database as buildings and other landmarks like roads do not change very often. The detected landmarks are then matched to a database of landmarks to establish the position. This method is more efficient as only the landmarks need to be matched based on their relationships between one another, rather than having to match individual pixels onto a large database of images.

2.2.1 Vision based Geo-location

Researchers from Linkoping University demonstrated a vision based navigation using geo-referenced information [1]. The vision based navigation system combines inertial sensors, visual odometry, and image matching of the on-board video to a geo-referenced aerial image. Visual odometry is used as a fast position update; however, this is subject to drift so the use of geo-referenced image registration is used for drift free position calculation. Image registration is the process of transforming different sets of data [1] into one coordinate system. In this work the correlation based image registration is used to match the sensed and reference image. This correlation based approach places each pixel of the sensed image over the reference image and based on a similarity criteria, it is decided which pixel location gives the best fit. The authors

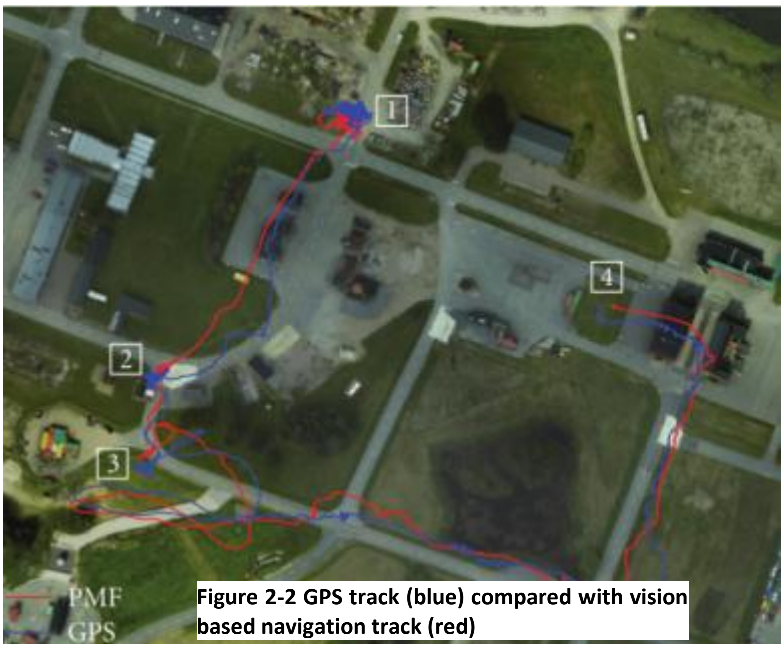


Figure 2-2 GPS track (blue) compared with vision based navigation track (red)

[1] In this case, the data is images of a scene that could be taken at different times, from different viewpoints from different cameras.

reported near real time performance of their correlation based image registration implementation; however the image registration step took several seconds. The advantage of correlation based matching is that it can be applied to images with no distinct landmarks. However, because this correlation based method is based on pixel intensity values, matching will be easily affected by differences between the reference and sensed images (e.g. seasonal changes like snow, different time of day).

In 2009 WU Liang and HU YunAn proposed a vision aided navigation method for aircraft based on road junction detection and GIS data[2]. This system addresses the problem of needing a very recent database of reference images for image matching. They overcame this problem by detecting road intersections in the aerial images. This means that you only need a fairly accurate database of road networks which is easily obtainable, and the road locations do not change with seasons, or time of day. Road junctions are also good reference landmarks because they are widely distributed in civilized areas, and are fairly distinct in aerial images. The basic principle that WU Liang used, is that the junctions are detected via a downwards looking camera on an aircraft, and these junctions are matched to a database of known junctions (from GIS) for the estimated position of the aircraft to be found. Two main methods can be used for road junction extraction, the first relies on road centre line crossing, and the other depends on geometric features. WU Liang and HU YunAn used a center line based road junction detection method which was first proposed by Steger in 1998[3]. Wu Liang did not test his algorithm over a wide variety of scenes.

Other researchers have investigated using vision based techniques for indoor navigation; due to the limited space available in indoor environments hovering platforms are always used. Researchers described a method for vision based corridor navigation using a small helicopter[4]. This method finds corner points along the corridor walls. The distance of the points along the floor of the corridor can be found by trigonometry using the height above the ground and camera properties of the helicopter. Keeping track of the distance between the helicopter and the corner points allows the system to predict the position and movement of the helicopter in the indoor environment. Similarly in 2008 [5] a quad-rotor helicopter platform was used for indoor navigation was developed. This method just tracks corner points in an image, and stores information about each corner in a database such as its location. Each new corner point that is found is compared to the points in the database, when a match is found the platform can estimate its position. The information from the onboard camera is combined with the inertial data onboard the quad-rotor platform to estimate vehicle motion.

Indoor techniques are based on a few assumptions of indoor environments (small structured environment, linear edges, slow movements) and are not always applicable to varying outdoor applications. However some of the fundamental techniques such as feature detection can be used.

2.2.2 Aerial Image Object Recognition

Since the area of vision based positioning in outdoor applications is fairly new, some more general vision based techniques that are applicable to vision based positioning will also be introduced There are many different approaches to detecting objects using computer vision techniques, this section will go over some of the most promising techniques relevant to object detection in aerial image are investigated.

In general a more reliable approach for vision based geo-location would be to detect static landmarks such as buildings. This will overcome the problem mentioned in [1] of having different reference and source images that could occur at different times of day, or seasons. For this related work about object detection will be useful to detect objects and landmarks to be matched to the database.

Building Detection

A hierarchical and contextual model for detection of objects was developed in [6]. This method focused on the overall scene, instead of only detecting a specific object (building, road, etc...). The basic approach was to devide the image into scene and object levels of detail. The two step algorithm first uses compositional boosting to detect roofs and roads. Compositional boosting groups edges into larger structures based on weak classifiers (groups) based on features (geometric and photometric). The authors reported that this results in a high number of false positives, and the second step of the algorithm prunes the false positives based on their local context in the image, using a top-down scene level component of the model. Approximately 200 hand labelled training images were used and taken from Google Earth. The results showed that using a two-step top down model allowed for reliable object detection, with a great reduction in false positives, however this process is computationally inefficient so is not suitable for real time applications.

Recently a novel method for automatic building extraction from aerial images was proposed in [7]. The method uses a hierarchical multilayer feature based image segmentation technique using colour. The segmentation is based upon the buildings geometrical features and context. A search tree is constructed using multiple levels of the mean shift segmentation algorithm so that the best image range resolution can be found for each region. The rooftop segmentation divides the image into potential rooftop regions based upon 5 constraints; total curvature, compactness, boundness prevention factor, decidability factor, and area. Finally the rooftop regions are validated by detecting the shadows. High accuracy results were obtained at 93%. This is a very accurate building detection method; however, the reliance on shadows for verification proposes difficulties to applying this to a geolocation system since the validation cannot be used in overcast weather. Once again, the computational requirements for this method are high taking a few minutes to process each image.

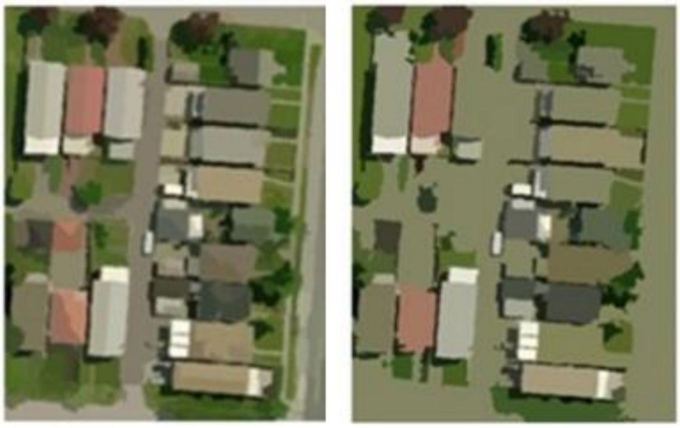


Figure 2-3 Multi-Level Segmentation

LIDAR Based Object Detection

An alternative method to detect objects in aerial images can be achieved with the use of Light Detection and Ranging systems (LIDAR). Researchers from the University of Melbourne investigated this problem[8]. They found it particularly difficult to obtain an accurate boundary of detected buildings using only LIDAR data so they combined both LIDAR and photogrammetric imagery. They used the LIDAR data to separate the ground level from a certain building height threshold, and then used the second region (building height threshold) to detect buildings using the aerial imagery based on colour information. Overall the combination of the LIDAR and imagery improve the results to a detection rate of about 94%, however the authors reported that there is still some inaccuracies building boundary accuracy.

Road Detection

There are two main forms of road detection, region based, or line based. Line based approached usually extracts edges of the roads; these edges are then combined to form road lanes. Region based methods find roads based on classification and shape criteria. Reliable road detection in open landscapes exist [9], however road detection in urban environments proves to be much more difficult due to the high complexity in urban aerial images. For this reason additional information is required such as colour, digital surface models, multi-spectral information to aid in the detection of roads or other objects.

As mentioned before, road intersection detection has been used for geo-location in [2]. However a more general another approach for detecting roads is presented in [10]. This approach is region based and has three steps. The first step is segmentation of the image into small segments. The second step is to group the small segments into regions that have a high probability to be a road section using shape criterion. The third step is to join together road parts, and remove road parts that do not match, in order to get rid of the false positives. This is done by looking at the direction of the road sections and joining the sections together. Colour infrared images were used, and this aided in the separation of vegetated areas and asphalt areas, based on the radiometric properties from the colour infrared images. Overall road sections were detected accurately; however, some difficulties were found when joining together the road parts. The authors reported processing time in minutes per image and are not suitable for real time applications

Haar Classifiers

Another method for object detection is through the use of Haar classifiers. Originally Haar classifiers were first suggested by Viola and Jones for use in face detection applications[11]. Haar classifiers match objects based on Haar-like features in the image as shown in Figure 2-4. These Haar-like features are essentially block like features in the image. The authors claimed that the detector is fast enough to run in real time with high detection rates. Further work on the Haar classifier in 2002 added angled haar-like features to further improve the performance of the algorithm.

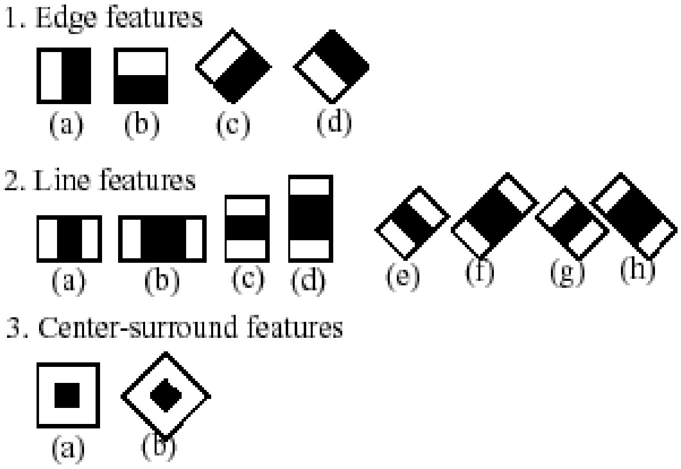


Figure 2-4 Haar-like features

Traditionally Haar classifiers have been used for face detection; however a few researchers have shown that it is possible to use Haar classifiers to detect other rigid objects such as body parts, cells, cars or bikes. The fact that the Haar classifier uses machine learning techniques means that to detect a new type object a dataset is required for the algorithm to train on.

A relevant implementation of using Haar classifiers for vehicle detection in aerial images onboard a UAV is described in [12]. The system automatically detects vehicles via the onboard camera on the UAV. The haar classifier is not very rotation invariant, so multiple classifiers are used at the 4 main orientations (0^{o},45 ^{o},90 ^{o},135 ^{o}). For each classifier angle, a training set of several hundred vehicle images at a similar orientation was trained.

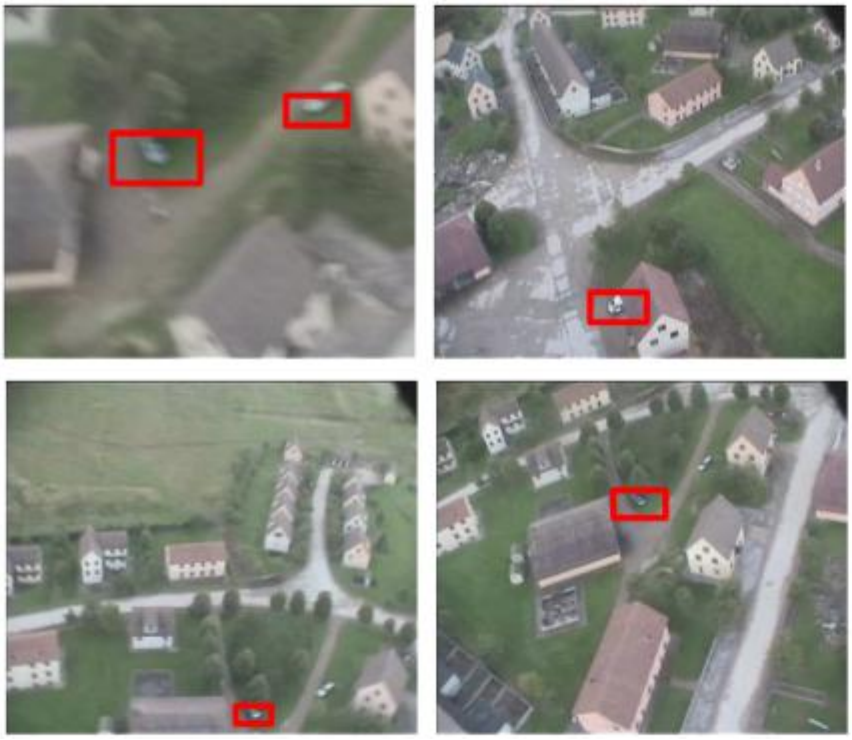


Figure 2-5 Haar classifier vehicle detection

Good detection rates were obtained using the haar classifier. The video was obtained onboard a UAV which is subject to vibrations and buffeting due to the wind, for this reason the video quality can degrade and become blurry, however as seen in Figure 2-5 the classifier is able to cope with blurry images, while still detecting the correct objects. The author used additional information from UAV sensors such as the altitude to filter out false detections of vehicles that did not match the expected size of vehicles for that altitude.

Speeded Up Robust Features

Another useful feature that can be used for object detection is the Speeded Up Robust Features (SURF) as proposed in [13]. SURF key points are ideal for detecting objects as each key point is both scale and rotation invariant. The authors compared the SURF detector to other common key point detectors such as the Scale Invariant Feature Transform(SIFT) detector.

A relevant application of using SURF based features scene matching is presented in [14]. The authors proposed using SURF as a faster method to match two images compared to image registration techniques. The first step it to extract all of the SURF key points and their descriptors from the reference image/map and store them in a database. When a smaller section of the image is loaded in real time, the SURF key points and descriptors on the input image are matched with those on the reference image and the location can be found. An example of this is shown in Figure 2-6. The authors reported that this can be done in real time. The SURF key points are matched a two tier approach. The first coarse match is based upon the bidirectional nearest neighbor method which removes a large portion of outlier matches. The second matching step is based on RANSAC and dominant line direction methods which find a much finer and accurate match between the remaining points.

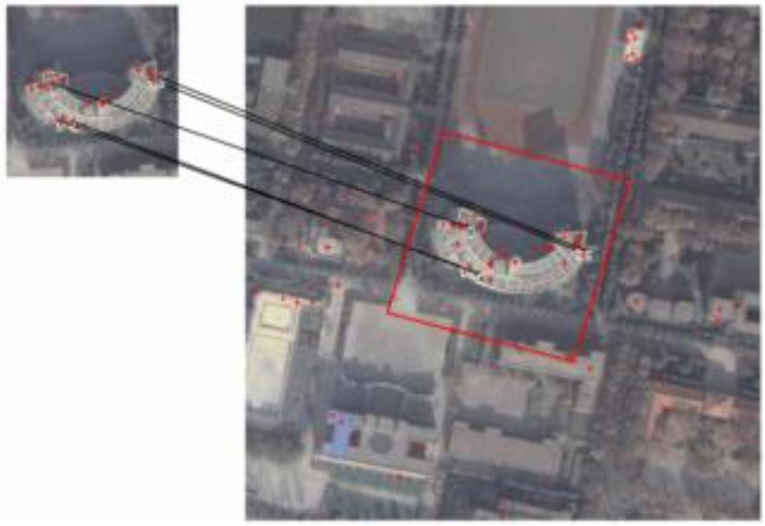


Figure 2-6 SURF matching

A performance comparison was also made to other image registration methods such as normal cross-correlation and Scale Invariant Feature Transform matching methods (SIFT). The SURF based detector outperformed all the other methods in terms of speed taking an average of 150ms to match an image. This method can be a possible solution for geo-location were a database of SURF key points of the flight area can be stored onboard the UAV. The key point detected on the onboard camera can then be matched to the database of featured to find the location. The other major advantage is that it can run in real time.

2.2.3 Vision Aided navigation with GIS data

A recent paper described a novel vision aided system that extracts object level features and matched them against data in a geographic information system (GIS)[15]. The advantage of using GIS data is that it is a digital representation of the world and requires less storage when compared to reference images. The first step of the algorithm is to estimate the position of the UAV based on previous knowledge and other sensor information, this will correspond to an initial search region in GIS. The GIS region in the database is simplified and used to construct a GIS model which is used to match features with the sensed image. The sensed image is processed to obtain visual geometrical features. Firstly, roads are detected using a fast linear structure detection and delineation method as previously described in the aerial object detection section. Once the roads are detected, the road endpoints, branch points and intersections are found using the FAST algorithm. Non maximal suppression is applied to reduce the number of false points. Finally the features extracted from the sensed image are matched to features in the GIS model by comparing the distance and angle between the points in the sensed image and GIS model. Overall this system performed well with the maximum time for a match taking about 10 seconds. However, this is still limited to higher altitudes, as when flying lower down very few if any road intersections will be found resulting in poor matching.

2.2.4 Conclusions

From the literature review most of the road and building detection algorithms are developed for GIS systems, or other mapping related problems such as remote sensing, or city planning. These applications typically do not require a high speed system and often can rely on powerful computers and offline processing. For this reason many of the algorithms are slow and complex. Many of the building detection algorithms also rely on shadow information to verify the location of a building, this works well for remote sensing applications were the images are satellite/ aerial which are typically taken during sunny days with little cloud cover due to the high altitude images. However, for a real time UAV system flying below the clouds, shadow information cannot always be used as weather can vary. Alternative methods to verify buildings will need to be found. Although real time object detection is not required, a fairly high speed update rate is required. Many of the results found in the papers also show that 100% accuracy for detection has still not been achieved, this should not be a large issue for the proposed geo-location system as it can cope with a few differences between the sensed and referenced images; however, it is still important to get reliable and accurate detection of features.

Most of the papers discussed focused either on urban, or rural areas for feature detection, this is because of the different nature of the problems. Urban environments are complex with many objects, and rural environments typically contain very few man-made objects.

The Haar classifier based object detection research is very promising as it is both accurate and fast, this will be suitable for further work to try and implement this system for detection landmarks with the downward looking camera. Detection of rigid landmarks such as buildings will be well suited to this algorithm. Landmarks that are to be detected in aerial images such as buildings and junctions are not always at the same orientation so the problem of detecting objects at multiple orientations will need to be addressed.

3 Image Registration using Speeded Up Robust Transforms

The first method to investigate is the use of image registration based techniques to match a given source image into a database of reference image. The proposed method is shown in Figure 3-1. This is a possible method for geo-location where the source image can be captured via a camera onboard the UAV, this source image can be matched onto a database of reference images. Once the source image has been matched to a reference image (real world locations of reference images are known) the position of the UAV can be determined. As mentioned before, trying to match images based on pixel level data such as cross correlation can be computationally demanding, such methods have been around for some time [16]. Cross correlation based image registration techniques are not feasible for use onboard a UAV for extended missions as the database of reference images will need to be very big, and searching such a large database will be very slow.

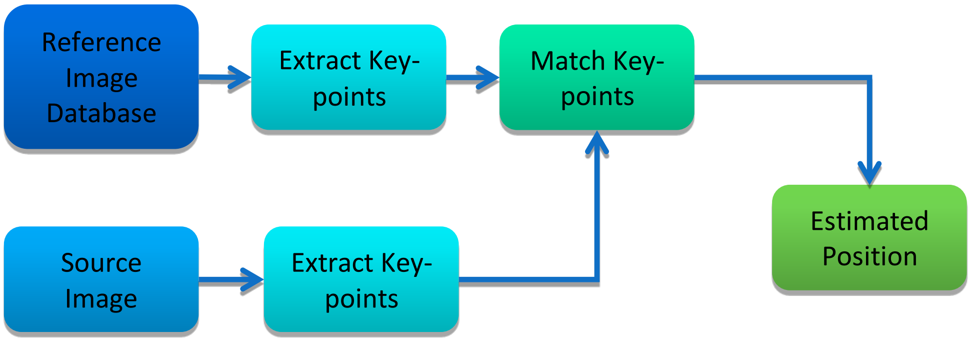


Figure 3-1 Image Registration Process

A different approach to image registration is described in this section. This new method will address some of the drawbacks of previous methods such as performance, and database size. The main advantage of using Image Registration techniques is that once a source image is matched to the database the location of the UAV is already known because the location of the reference image is known.

3.1 Speeded Up Robust Features

Instead of trying to match images based on a pixel level, a feature/key point based method will be used. This method essentially picks out unique points on an image and then describes them. The key points are matched instead of the each pixel which greatly improves performance. This thesis makes use of the Speeded Up Robust Features key point detector which is widely used in computer vision applications. There are many other key point detectors, the most notable being SIFT (Scale Invariant Feature Transform) [17] which has seen wide use. The SURF detector was chosen as it has similar performance but is much faster.

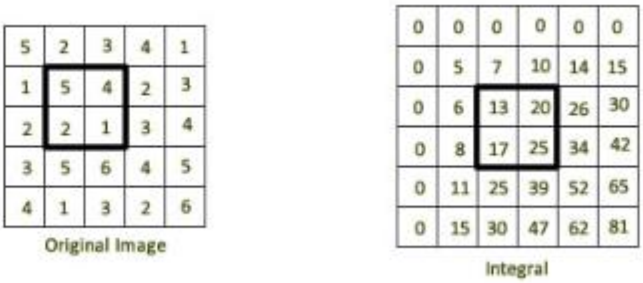
The Speeded Up Robust Features (SURF) detector and descriptor was first introduced by Herbet Bay et al in 2008 [13]. The SURF detector is a scale rotation invariant feature detector and descriptor which has found many applications in computer vision in particular object detection due to its repeatability and efficiency.

3.1.1 Integral Images

The SURF detector is based on Hessian matrix approximation. This allows the use of integral images which greatly improves performance. An integral image (summed area table) quickly and efficiently generates the sum of values in a rectangular sub-region of an image. The approximated sub regions allow of much faster processing of the images as less computational instructions are required for each region. The value at any point (x,y) of the integral image, I is the sum of all the pixels above and to the left as shown below.

$$I(x,y) = \sum_{\substack{x^i \leq x \\ y^i \leq y}} i(x^i, y^i) \tag{1}$$

Below in Figure 3-2 Integral Image shows a visual example of an original image on the left in terms of pixel values, and its corresponding integral image on the right.



5	2	3	4	1
1	5	4	2	3
2	2	1	3	4
3	5	6	4	5
4	1	3	2	6

Original Image

0	0	0	0	0	0
0	5	7	10	14	15
0	6	13	20	26	30
0	8	17	25	34	42
0	11	25	39	52	65
0	15	30	47	62	81

Integral

Figure 3-2 Integral Image

The key feature of integral images is that the integral image can be computed in a single pass over the image. Finally the summation of pixels in any sub region of the integral image is the difference between the top right pixel and the bottom left pixel. This allows rapid summation calculations in rectangular regions of the image which is particularly well suited to the features used in the SURF detector.

3.1.2 SURF Key Point Detection

The SURF key points are based on a Fast Hessian detector, which gives a good balance between computation time and accuracy. The detector can be described as the following
Given a point X in a image *I* , *X = (x,y)* the corresponding Hessian matrix H, at scale σ is defined as:

$$H(X,\sigma) = \begin{bmatrix} L_{xx}(X,\sigma) & L_{xy}(X,\sigma) \\ L_{yx}(X,\sigma) & L_{yy}(X,\sigma) \end{bmatrix} \tag{2}$$

Were L_{xx} represents a convolution of the Gaussian second order derivative of the image I at point X. The Gaussians are known to be optimal for scale-space analysis and this is one of the reasons why the SURF detector has scale invariant properties. In order to improve performance of the detector the authors approximated the second order derivatives using box filters as shown in Figure 3-3 using 9x9 box filters.

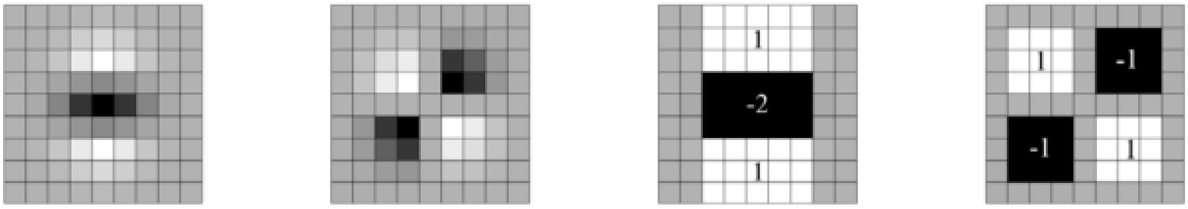


Figure 3-3 Left: Second Order Gaussian Derivatives, Right: Approximated Box Filter

The box filters enable the use of Integral Images (as described previously) which dramatically speed up the convolution and computation time. The convolution of each box filter and image can be denoted as D_{xx}, D_{yy}, D_{xy}. Using the box filters the approximated Hessian Determinant becomes:

$$\det(H_{approx}) = D_{xx}D_{yy} - (0.9D_{xy})^2 \qquad (3)$$

Scale space relationships are usually implemented using image pyramids. In order to do this the images are repeatedly smoothed using a Gaussian function, this is known as a low pass pyramid. This smoothing is done using integral images and box filters for computational efficiency. The pyramid of each different scale is constructed by increasing the box filter window size. Once the Hessian matrix determinant has been approximated at each scale, non-maximum suppression is applied around the neighborhood to find the maxima. The maxima points are then interpolated in both scale space, and image space which will give stable points. Each stable point is considered to be an interest point/ key point.

3.1.3 SURF Key Point Description

Once a set of key points are found in a given image, each of these points need to be uniquely described so they can be matched as described later. The main description feature of each key point is the dominant orientation. By taking into account the orientation for each key point, allows the SURF detector to be rotation invariant.

In order to find the orientation of each key point, the haar wavelet responses in the X, and Y direction for the point and in a neighborhood around the point are found. An example of haar wavelets and SURF key point are shown in Figure 3-4. Again integral images are used for fast computation of the Haar wavelets.

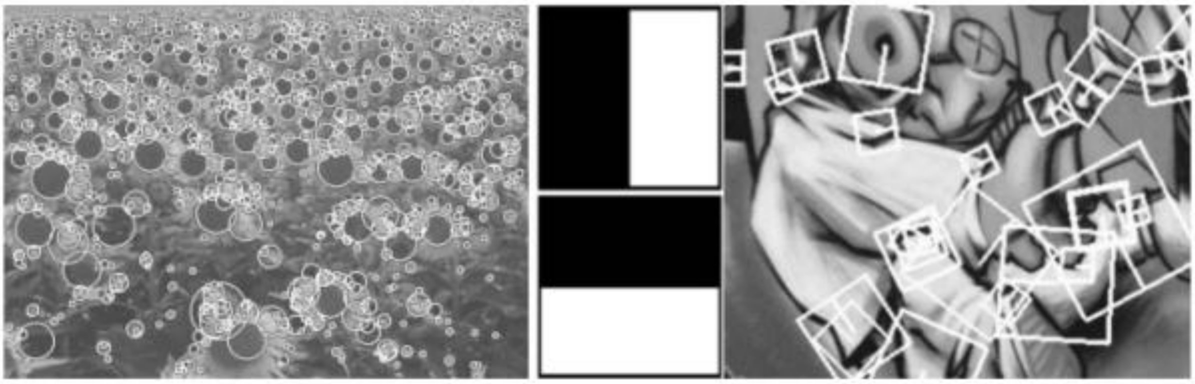


Figure 3-4 Left and Right: SURF Keypoint in images, Middle: Haar wavelets in X, and Y

The Haar wavelets are calculated in a circular radius of 6s (s is the scale of key point) around each key point. The haar wavelet responses are represented as vectors were the horizontal response strength is plotted along the horizontal axis; the vertical response strength is plotted along the vertical axis as shown below in Figure 3-5.

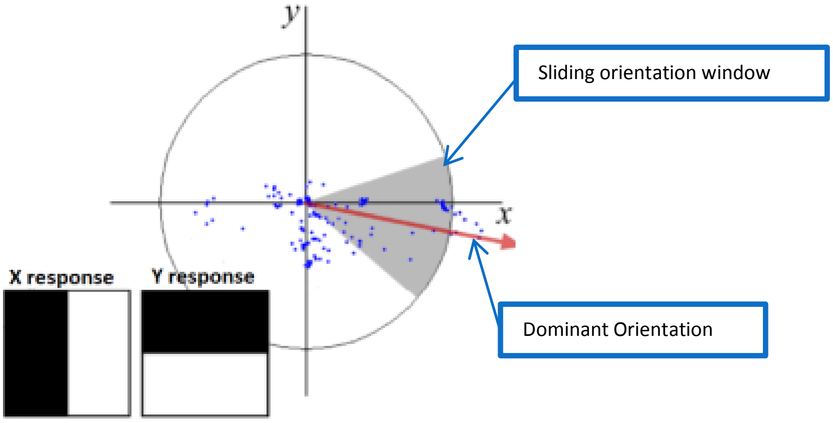


Figure 3-5 Haar wavelet response and orientation

The dominant orientation of the key point is found by calculating the sum of the horizontal and vertical responses over a sliding window of 60^{o}. The largest vector found across all the sliding windows is the dominant orientation of the current key point.

The dominant orientation of each key point is not sufficient to uniquely describe each key point. For each key point a 20s (were s is scale of key point) square region is created that is orientated along the dominant direction previously calculated. This square region is further divided into 4x4 sub regions. In each sub region, the X and Y Haar responses are again calculated and summed. Using the Haar responses, a 4D description vector V is formed:

$$\mathbf{v} = \left(\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|\right) \tag{4}$$

This results in a descriptor that has a length of 64 for each key point $4 \times (4 \times 4) = 64$. An illustration of this can be seen in Figure 3-6

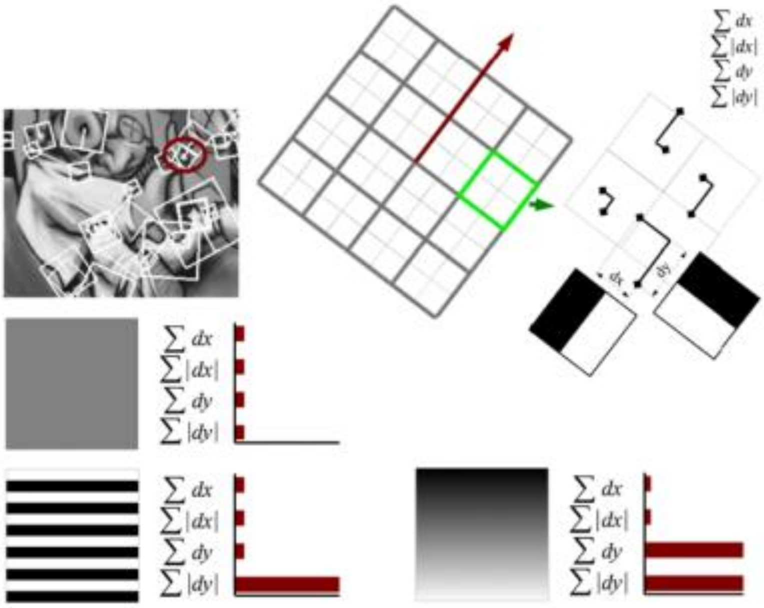


Figure 3-6 SURF feature description

3.2 Matching Features

As described previously, key points are found in an image they are each uniquely described. In this section describes how these key points can be matched between a source image, and a reference database of key points to enable localization of a UAV flying over a known region.

The SURF detector includes a fast indexing step which is very useful for improved performance during matching. The trace of the Hessian matrix for each interest point is included in the descriptor. For a typical interest point which is generally blob type structures, the trace easily distinguishes bright blobs on dark backgrounds and dark blob on light background. This trace was previously calculated during the key point description stage and can be included at no additional computation cost. The main advantage of using such a method enables faster matching as only features containing the same contrast are considered as potential matches. Figure 3-7 shows an example of two features that will not be considered for matching as the contrast is different.

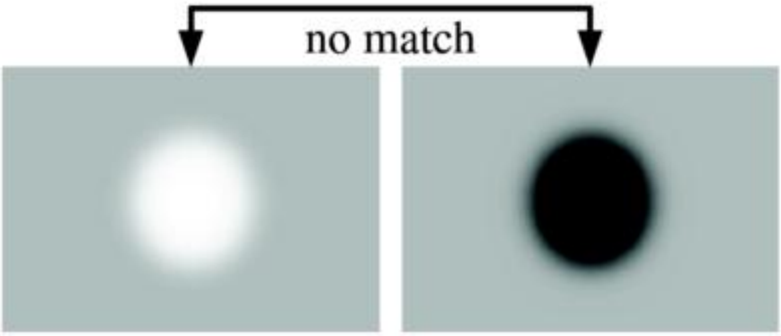


Figure 3-7 Different contrast features are not considered for matching

3.2.1 Brute Force Matching

The first method of matching that is considered is a brute force matching method. The brute for matching method takes each feature and compares is to all the other features of similar contrast. Brute force matching, although simple to implement can be slow when comparing over a huge amount of features which, would be the case if the reference database is large.

3.2.2 FLANN Matching

The Fast Approximate Nearest Neighbors (FLANN) matching method was first presented [18] as a matching method that is claimed to be an order of magnitude faster than previous methods. The FLANN matching is based upon the K-nearest neighbors search. The main advantage of using nearest neighbor search is that only a few key point that are in the neighborhood are considered, greatly reducing the number of comparison operations when compared to brute force matching.

3.3 Testing

In order to compare the performance of the SURF detector for vision based positioning of a UAV a series of test were performed. The tests assumed a level flight condition of around 800m AGL which is a typical cruise condition.

3.3.1 Reference Database

To create the database, a large database of reference aerial imagery is obtained from the publicly available Google Earth database. For initial tests a small region in Milton Keynes was extracted and all the SURF key points were extracted and stored in the database along with their location on the map. In order to save computational time the matching will not search over the entire database, but based on previous position estimates a smaller sub region will be used. This sub region will constantly move based on IMU data and previous position estimates of the UAV. A section of the reference database and the corresponding SURF key points is shown below in Figure 3-8 and Figure 3-9. The reference database was taken at an approximate altitude of 2000ft which would be a typical flight altitude, this corresponds to a spatial resolution of 0.9m per pixel.

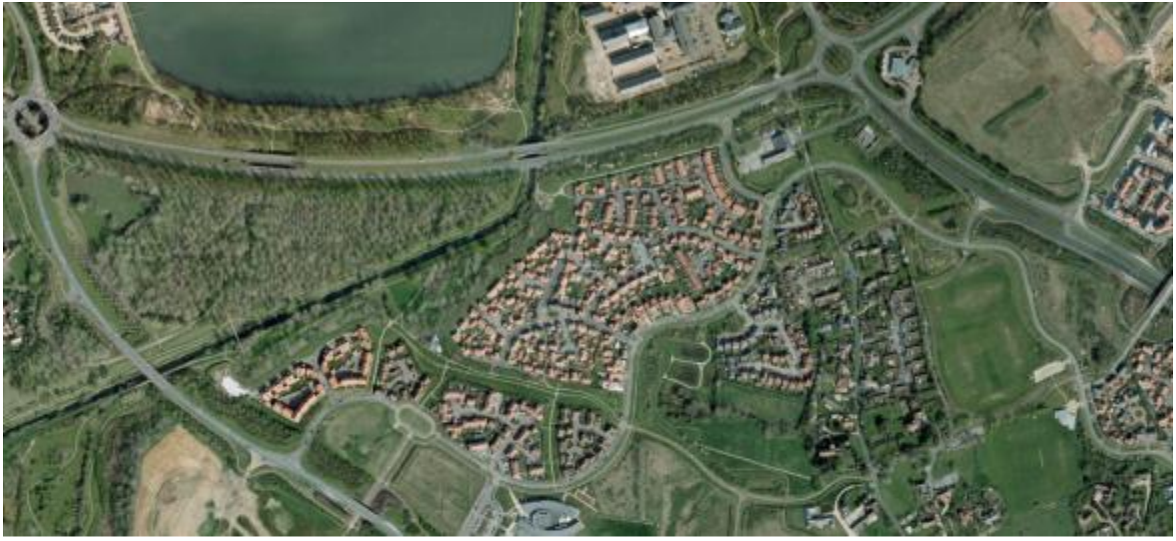


Figure 3-8 Section of Reference Database

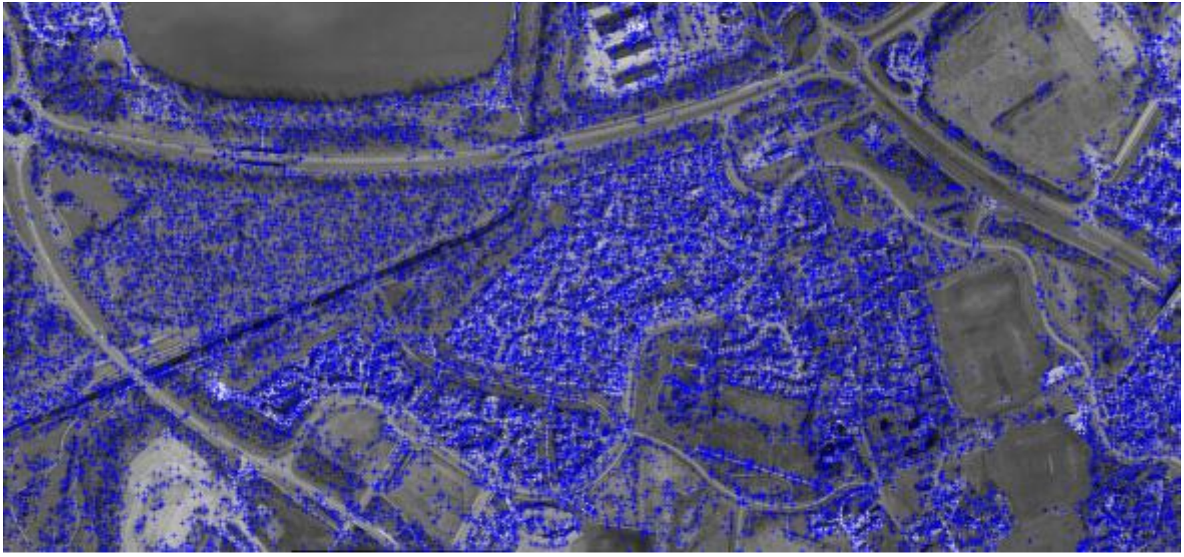


Figure 3-9 Reference Database SURF Key Points

You can see that in some regions that are sooth, little or no SURF key points are detected. When flying over regions that are smooth such as large bodies of water, this method will not work.

3.3.2 Key Point Matching Tests

The two matching techniques used in this thesis are Brute Force Matching, and FLANN Matching as described previously. In this test the two methods will be compared in order to find the most suitable matching method for use on a vision based positioning system.

Performance

The performance of the key point matchers will be compared in order to establish which of the two matching methods are more efficient in terms of computation time. To do this the matching time will be compared for an increasing amount of features.

For this test a few reference images was matched to a series of feature database that had an increasing number of features. An example of a reference image used in one of the tests is shown below.

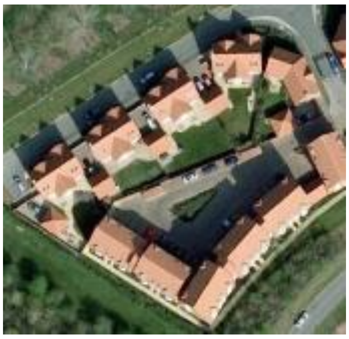


Figure 3-10 reference image with 476 SURF key points

Figure 3-10 had a total of 476 SURF key points. An example of the matching result is shown below in Figure 3-11. This is a typical scenario that the vision based positioning system will encounter, were the smaller image on the left is what is seen by the onboard camera. The larger image on the right would represent a search area based on were the aircraft is estimated to be based on previous measurements. The features from the source image onboard the aircraft are extracted and matched to the features in the database to establish a position.

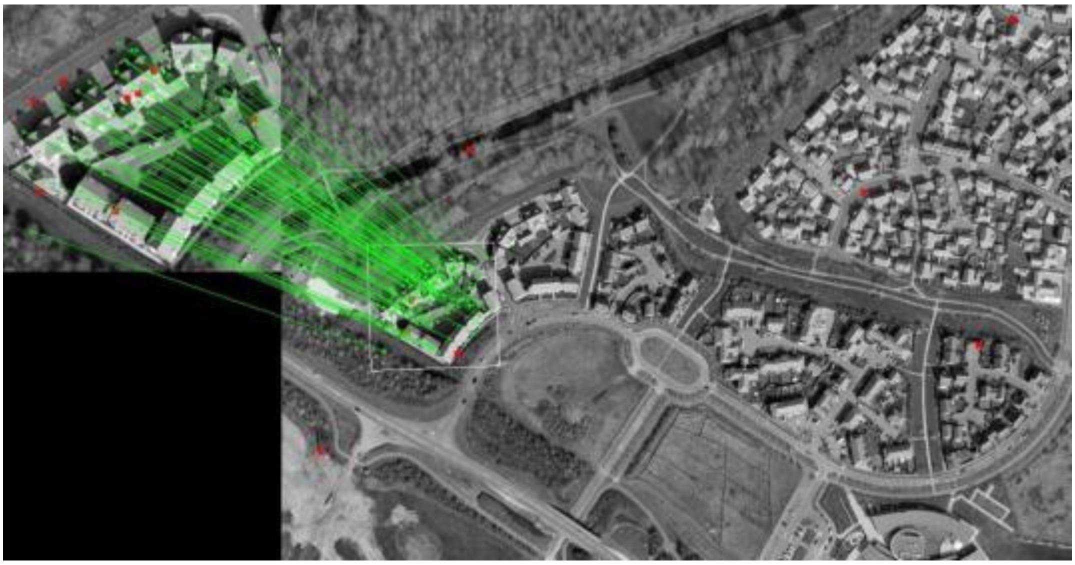


Figure 3-11 SURF key point matching

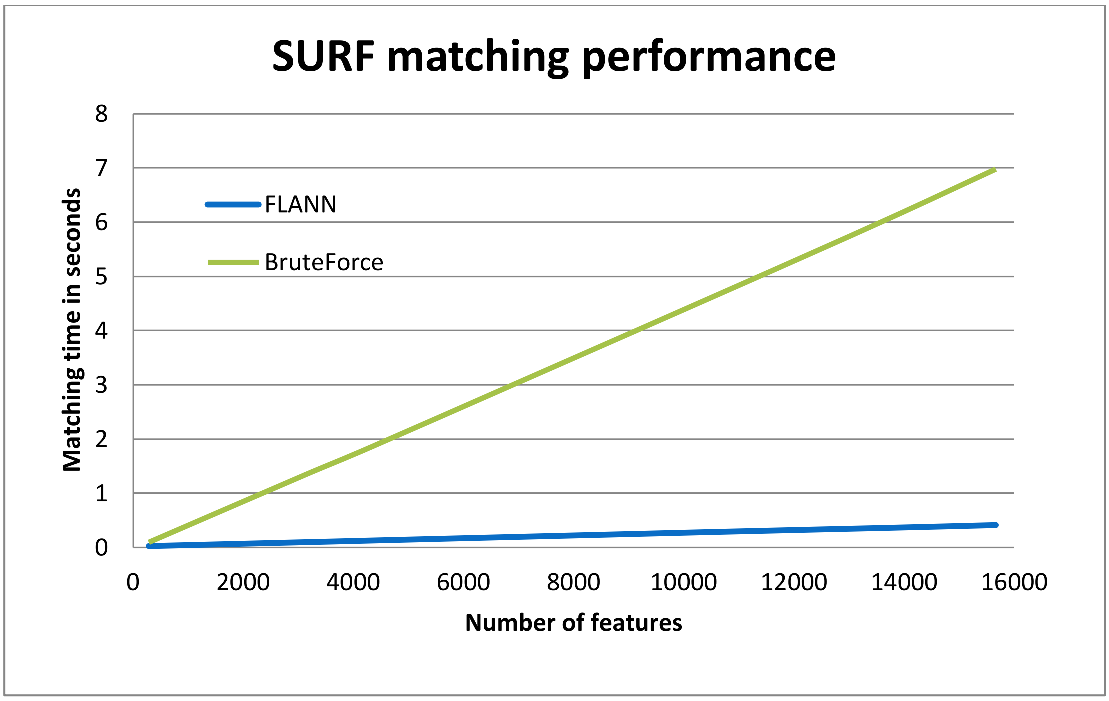


Figure 3-12 SURF matching performance between Brute force and FLANN matching

It is clearly visible in Figure 3-12 that the FLANN based matching outperforms the brute force matching method in terms of computation time. When there are few features to compare (up to 600) the two matching methods have similar computational time. However as the number of features increases, both the matching methods take longer to complete matching, this is fairly obvious as more features need to be compared. As shown in the figure above, the brute force matcher is much slower than the FLANN matcher for large amount of features. This is mainly due to the nature of the brute force matcher that directly compares each feature of the source and reference image. The FLANN matcher is much more efficient for larger features as this method looks at feature space and only compares features that are deemed to be within the neighborhood of the current feature. It can be seen that as the number of features increases the FLANN matcher also takes longer for computation because more features need to be sorted into feature space, however the FLANN matcher outperforms the brute force matcher in terms of computational time for high numbers of features. Typical for visual based positioning system the number of features to compare will be large, and therefore the FLANN based feature matcher is the most suitable method to use. The data in Figure 3-12 was obtained using the OpenCV python library, on a 2.4Ghz Intel Core 2 Duo processor.

3.3.3 Scale/ Rotation Tests

Testing the rotation and scale effects will enable a more suitable reference database to be constructed. The SURF detector is a scale and rotation invariant detector so it should be able to handle conditions at different scale and rotations. However the main purpose of this test is to find the maximum difference altitude between the reference and source image that the system can cope with. This will establish the ideal resolution of the reference database. Typically at higher altitudes the spatial resolution[2] of image reduces.

[2] Spatial resolution is the corresponding ground on the Earth surface size of a pixel in an aerial image.

Scale

The test will establish effects of spatial resolution on matching performance. The reference database will use aerial images of a built up environment at an altitude of 2000ft which would be a typical mission altitude. This corresponds to a spatial resolution of approximately 0.9m, were each pixel on the image corresponds to a distance of 0.9m on the ground. The tests will attempt to match features in steps of 300ft above and below the reference altitude until no match has been found at either end. This was repeated over several regions and with different images. The averaged results across all of the images are presented below.

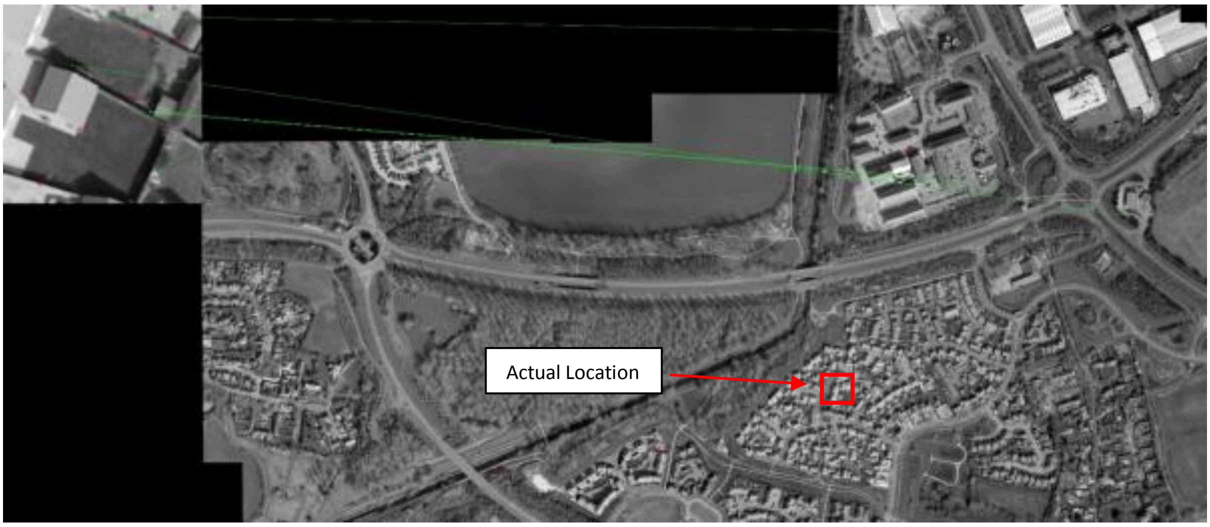


Figure 3-13 No match found at high spatial resolutions

As shown in Figure 3-13, high spatial resolutions do now make for accurate matching. The source image at lower altitude has a much greater spatial resolution than the reference database. There are much fewer correlations between the two as the reference database has a lower detail of features and therefore many features are missed. Because of this no match is found between the two.

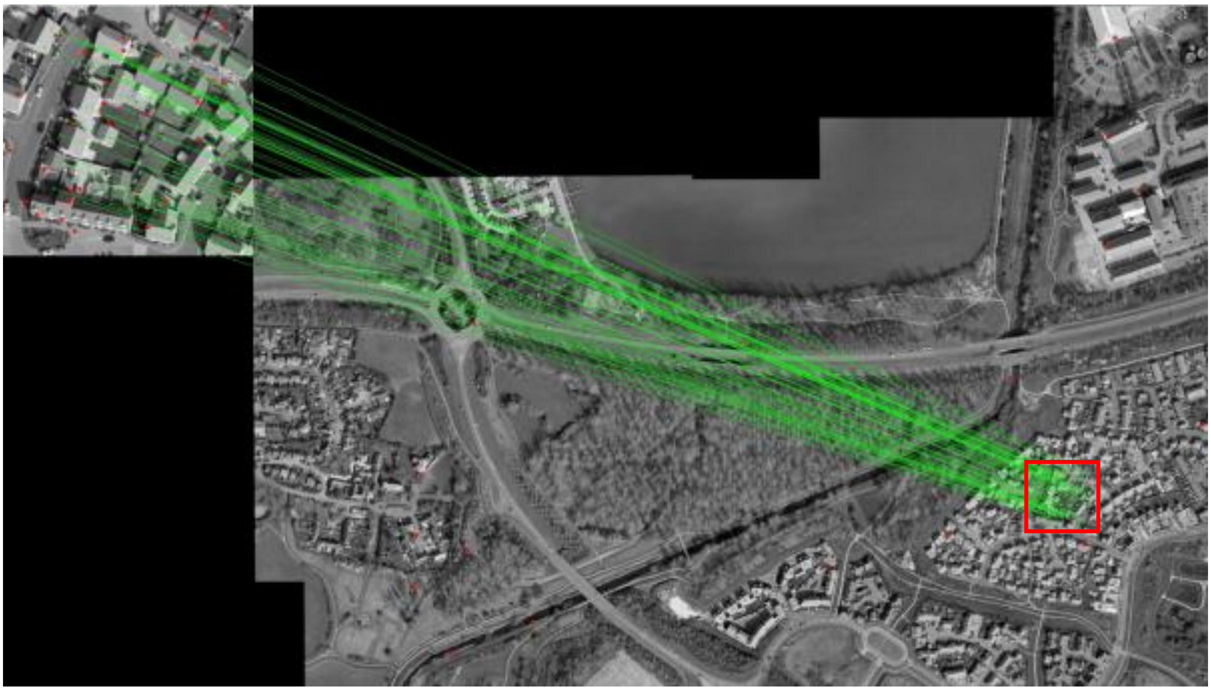


Figure 3-14 Match found at spatial resolution of 0.4m

At higher altitudes the reference database has more features present that are source image, and therefore a match can be established.

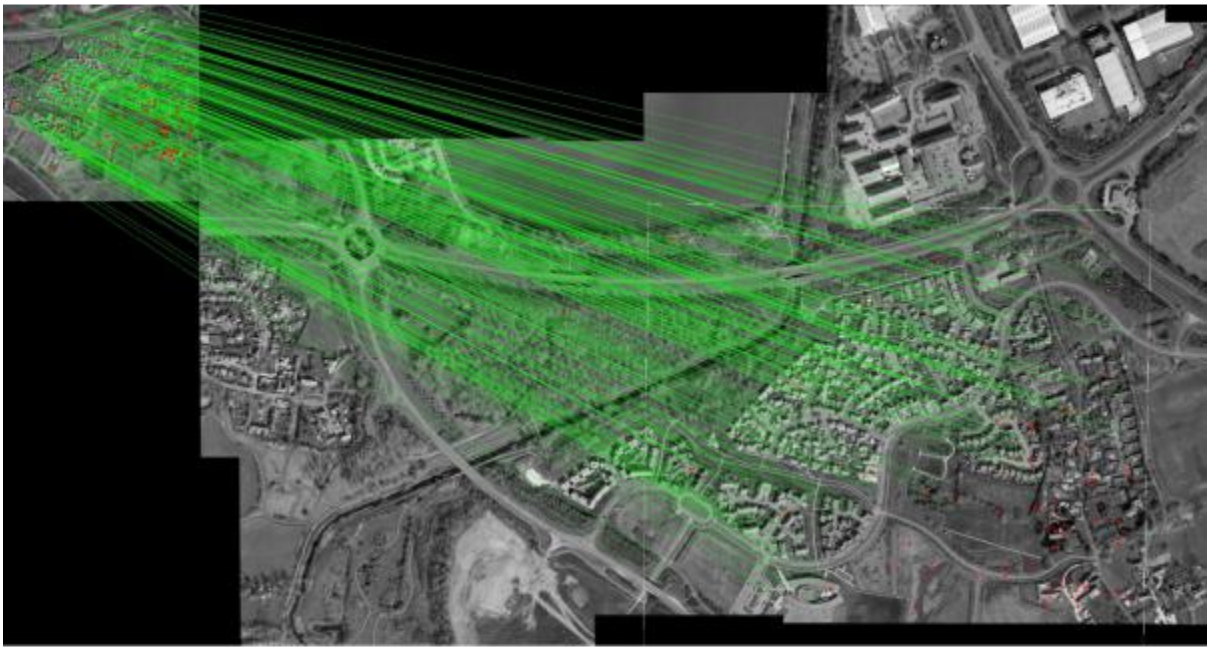


Figure 3-15 SURF Match found at spatial resolution of 1.6m

Some of the results are shown above for different spatial resolutions. The test was conducted over a few different regions with the reference database having a spatial resolution of 0.9m. Matching was compared with source images with spatial resolutions higher and lower than the reference database. The results are presented in the table below.

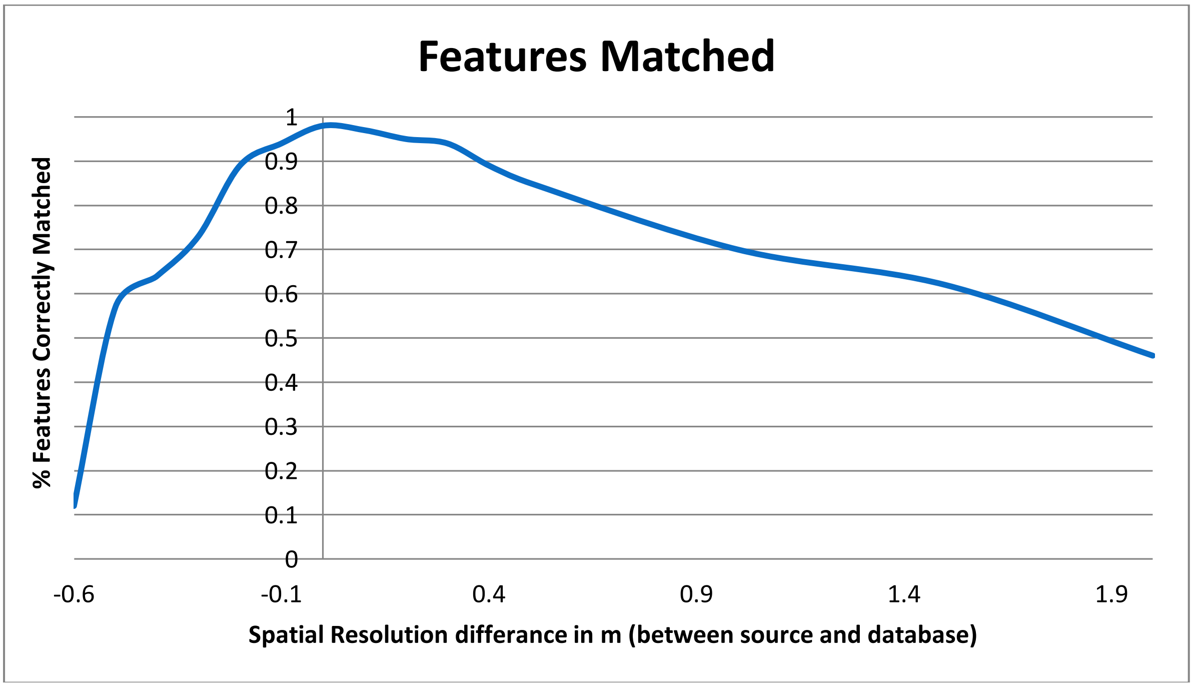


Figure 3-16 % of SURF key points matched at different spatial resolutions

As shown in the figure above, a range of spatial resolutions were tested. The highest rate of matching occurs 98% when there is the smallest difference of resolutions between the source and reference images. However as the spatial resolution difference gets larger the matching performance breaks down. This is more noticeable on the left of the graph for spatial resolutions that are higher (lower altitude) than the reference database. The main reason for this is that each feature on the source image contains additional noise due to the added detail of the higher spatial resolution. The other factor that causes problems at higher resolutions is the projected ground coverage in the images. At lower altitudes the source image only covers a small region of the reference database; this means that there are fewer possible matches between the source, and database. The presence of added noise at higher spatial resolutions combined with fewer features means that performance at low altitude is poor.

This problem is less noticeable at lower spatial resolutions (higher altitude) as the source images contain less noise than the reference database. This means that a match can occur for altitudes that are much higher than the reference database.

It is also important to note that only about 40% of the features are required to be matched to establish the location of the source image on the reference database.

Perspective/rotation

The SURF key point detector is claimed to be rotation invariant, we will perform tests to verify this claim on aerial images, and will also test perspective differences. The reference database assumes perfect top down aerial images, however when flying onboard an aircraft the camera may not always be perpendicular to the ground. This test will try to detect regions when the source image has some perspective error.

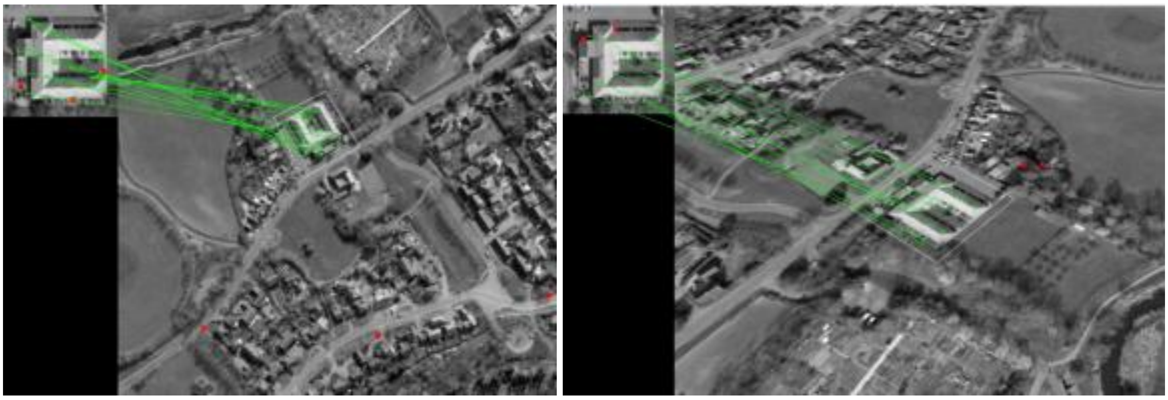


Figure 3-17 SURF rotation and perspective tests show matching can occur at different perspective and rotations

As shown by the images above, the SURF key point detector is capable to deal with rotation, and also slight changes in perspective. The quick test that was performed over a variety of regions has confirmed the claims of the SURF detector being rotation invariant, but has also shown that the detector can also deal with simple perspective errors too.

3.3.4 Robustness Tests

Previous methods of image registration noted difficulties when the scene changes due to different lighting conditions or seasons. This test will investigate the robustness of the SURF key points under different source and reference image conditions.

This test will attempt to match a smaller sub region of the reference map onto the complete reference map. This smaller sub region will emulate that view seen by a camera onboard the uav. The accuracy is measured in terms of correctly matched SURF key points between the source image and reference map.

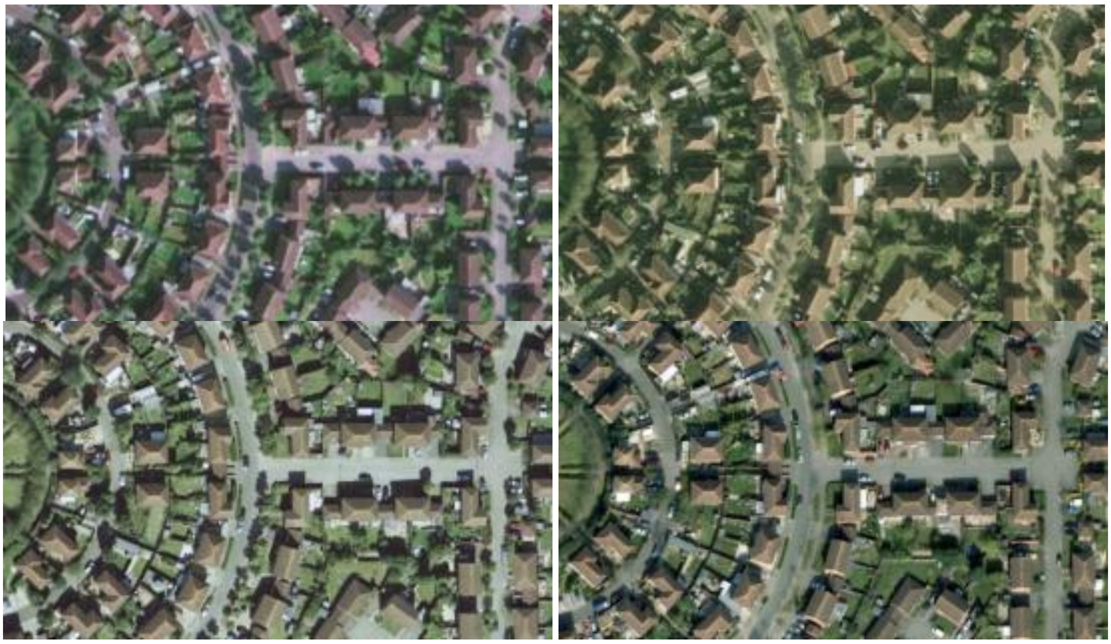


Figure 3-18 Aerial images taken at different dates

As shown in the images above, you can see how a given region may look different when the images are captured in different conditions and different cameras. This test will establish the robustness of the SURF detector to cope with such changes.

In this test the source and reference images will have the same spatial resolution. The only difference will be the conditions of the source image. The reference database was created using good earth images that were obtained in 2009.

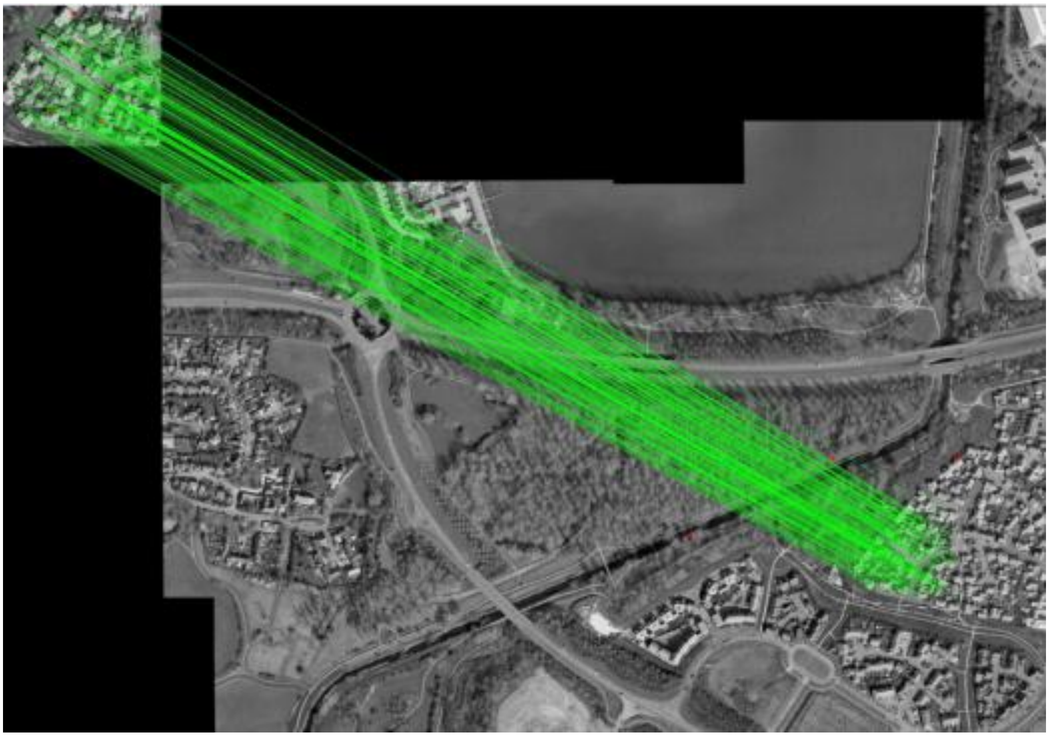


Figure 3-19 correct matching with same conditions in source and reference images

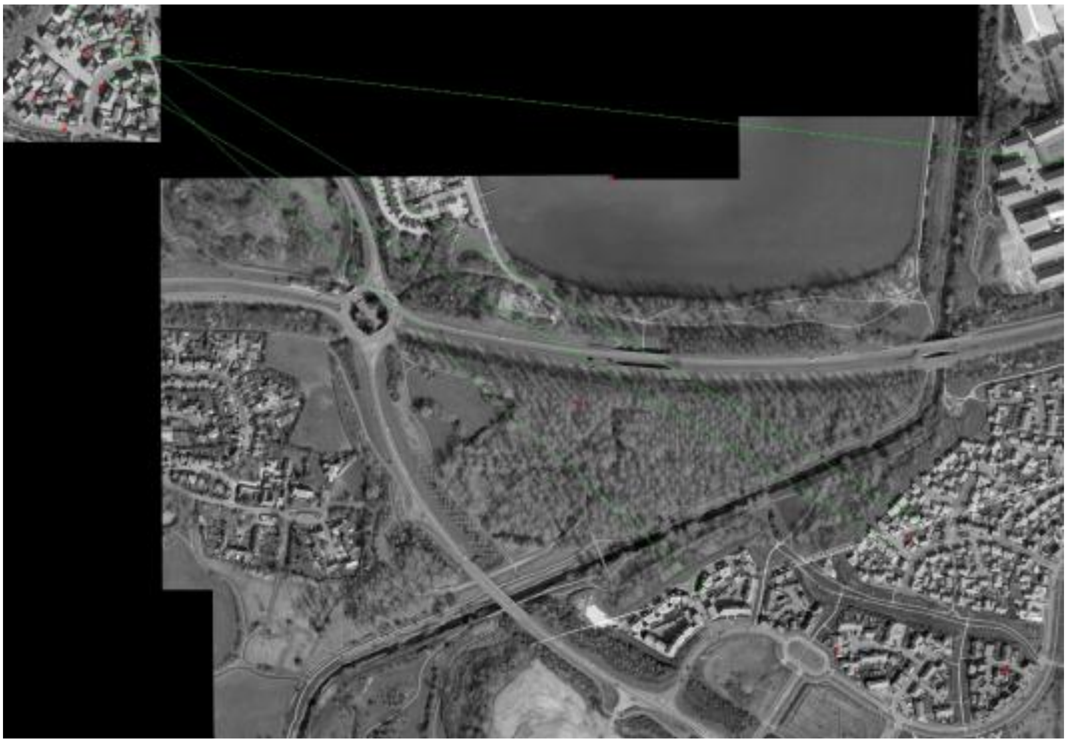


Figure 3-20 Poor matching with different conditions

However as shown in the image above, when the conditions change very few features a successful match cannot be established even though some features are correctly matched. Despite the images being very similar, the change in conditions between the two result in a poor match.

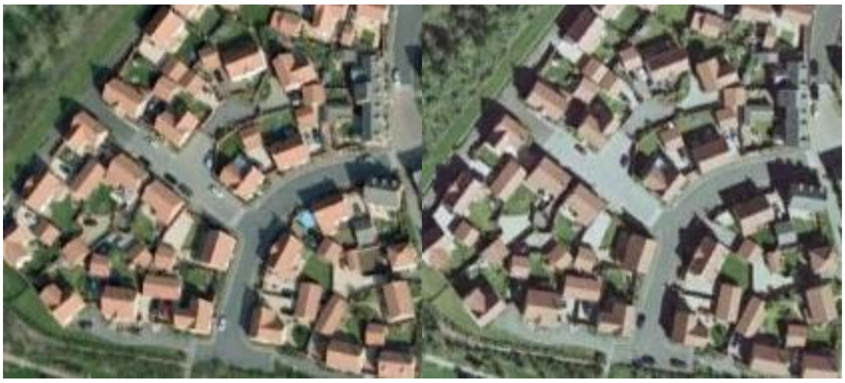


Figure 3-21 Slight condition changes between the same region (reference image left, source image right)

The robustness tests show that this method will either require up to date reference images, or a reference feature database that stores multiple image conditions. Post processing such as histogram equalization was used to try and balance the brightness and colours between the source and reference images, but this did not have any significant improvement on the matching performance. Both these solutions are not ideal, main because they do not deal well with changes between the source and reference images. For this reason the rest of the thesis will focus on using higher level object level detection methods. Having to rely on upto date reference images is possible, however it is not a practical method to rely on for navigation as reference data would have to be obtained at regular intervals for a mission area do deal with weather and seasonal changes, this would result in a very large storage requirement as well as greatly limiting the overall capability of a vision based navigation system.

4 Edge Based Building Detection

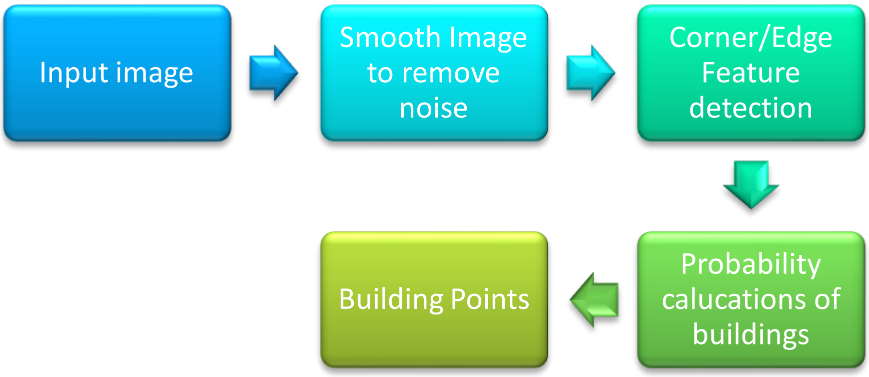
Previous methods investigated are all feature based methods, where features are matched between a reference and source feature set. As shown in chapter 3, high speed localization can be achieved, however when the reference and source images have large differences due to lighting, or seasonal changes the matching performance breaks down rapidly. In order for accurate vision based positioning, up to date reference images are required. The requirement of regularly updated reference images before each mission may not always be feasible. To overcome this problem, higher level object based matching approaches are investigated in the rest of this thesis. Key objects such as buildings, and road intersections are generally constant in appearance and will be visible even when the reference and source images are taken at different times of day/seasons. Overall the objects may look different, but there are common features of each object such as the rectangular shape of buildings and roads. The future chapters in this thesis investigate different approaches for robust object detection in aerial images.

If we look back at a previous image in chapter 3 Figure 3-21 (previous page), you can see that landmark features like buildings, and road intersections are the same irrespective of lighting or seasonal changes. The colours may vary between seasons, however the key features like corners and lines will remain constant as landmarks like buildings and roads do not change. Over a long period of time (years) the buildings and roads might change since new buildings are built or modifies, new roads are created, however these updates are much less frequent than that requires of the previous methods. Also information like the buildings and roads are stored on GIS databases, and are much easier to update compared to having to obtain a new set of aerial images for a given region.

The methods investigated in the rest of this thesis will focus on detecting buildings and roads intersections under a variety of different conditions. The basic idea is to detect the landmarks froma given image obtained by the UAV, and then match then to a database of landmarks on a GIS database to establish a position. The GIS database allows for a much more efficient method of storage, and access allowing for a larger area to be covered, and also is easier to update and maintain.

4.1 Finding Buildings

It can be agreed that corners and edges are the most constant features that define a building in an aerial image. A method is proposed that detects the edge and corner features in a given aerial image, and based on this information a decision will be made to decide if a group of corners and edges give enough evidence for a building. The process used to find buildings in an aerial image is shown below


Input image
Smooth Image to remove noise
Corner/Edge Feature detection
Probability calucations of buildings
Building Points

4.1.1 Image Smoothing

Aerial images that are high resolution contain a lot of smaller details, in particular small objects. These small objects are often detected as edges/ corners and make it more difficult for accurate building detection. For this reason the image undergoes some smoothing in order to remove some of the noise in the image. A bilinear filter was applied to the image to remove most of the noise, but still keep the key edge information, allowing for more accurate edge/corner detection around buildings. This can be shown in figures 1-1, and 1-2 below where the bilinear filtering results in more accurate corner detection around buildings.

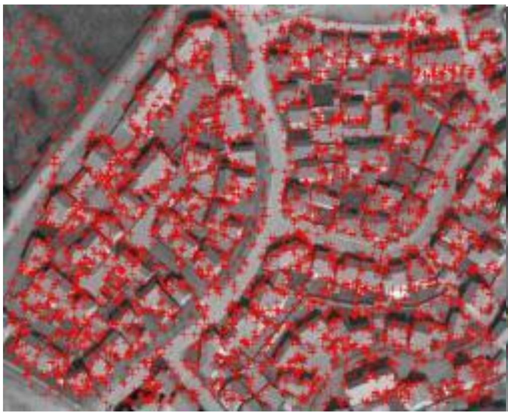


Figure 4-1 Garbor corners, no filtering

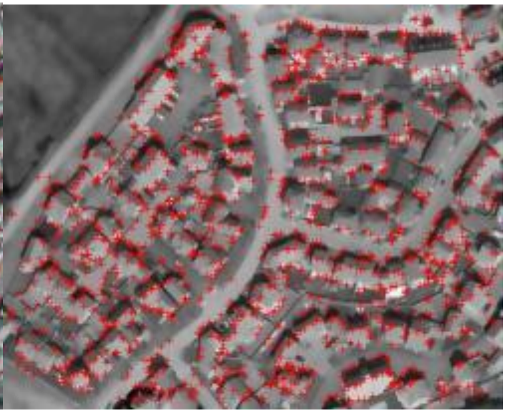


Figure 4-2 Garbor corners, bilinear filtering

4.1.2 Feature Detection

Harris Corner Detector

The first detector used is the Harris corner detector, as described in [19]. Essentially the Harris corner detector consists of three steps. The first step is to calculate the gradients in x, and y using smoothed Gaussian filters. The second step is the form the matrix of the gradient responses. The final step is the calculation of the eigenvalues for the matrix. If both eigenvalues have large positive values the point is a corner.

FAST corner Detector

The FAST corner detector uses machine learning techniques in order to find corners in an image [20]. Essentially for each candidate pixel, 16 neighbors around the candidate corner pixel are checked to see if at least 9 connected pixels that pass a series of criteria checks. If at least 9 neighbor pixels meet the checks, the candidate pixel is marked as a corner. Tests are performed using machine learning techniques in order to improve performance.

Garbor Filtering corner Detector

The Garbor Filter based corner/edge detector was considered[21]. This linear filter is obtained by multiplying a harmonic function with a Gaussian function applied at different frequencies and orientations to extract useful features from an image. The unique feature of the Garbor filter is that it has real and imaginary components, each representing orthogonal directions.

4.1.3 Building Probability

The more corner or edge features that are found in a group generally increase the chance of a building being present there. Therefore a probabilistic logic based approach is used to detect buildings based on the detected corner features. The detected corner features represent observation, and their probability density function is estimated, from this the building locations can be found.

For each of the detected features (corners), the gradient direction, and the magnitude is calculated. The gradient direction for the detected feature point can be found as:

$$O(x,y) = arctan\left(\frac{I_y(x,y)}{I_x(x,y)}\right) \tag{5}$$

Where I_y and I_x are the smoothed gradients in x, and y for an image $I(x,y)$which are found by applying smoothed Gaussian filters.
The magnitude can be calculated as:

$$M(x,y) = \sqrt{I_x^2(x,y) + I_y^2(x,y)} \tag{6}$$

A weighting is applied to each of the corner features, this weighting is calculated by finding the connected pixels around the corner feature with a similar magnitude. The sum of these pixels is the weighting of the corner feature. Strong corners will have a larger area of similar gradient magnitude and therefore a stronger weighting, opposed to weaker corners.

Kernel Density Estimation

In order to estimate the probability density function, a non-parametric method is used as defined by Silverman[22]. The estimated kernel based probability density function $p(x,y)$ is defined by:

$$p(x,y) = \frac{1}{nh}\sum_{i=1}^{n} N\left(\frac{x-x_i}{h}, \frac{y-y_i}{h}\right) \tag{7}$$

Where h is the smoothing parameter (also known as the window width)
The observations are represented by (x_i, y_i) for $i = 1, \dots \dots, n$

The kernel function $N(x,y)$ should satisfy:

$$\sum_x \sum_y N(x,y) = 1 \tag{8}$$

For a given aerial image it is not possible to determine how many buildings there are to be detected, therefore it is important to use a variable kernel function. To do this the local density of each observation smoothed by a scale parameter σ_i. Therefore the new probability density function is given by:

$$p(x,y) = \frac{1}{nh}\sum_{i=1}^{n} \frac{1}{\sigma_i} N\left(\frac{x-x_i}{h\sigma_i}, \frac{y-y_i}{h\sigma_i}\right) \tag{9}$$

Where σ_i is the variable scale parameter for $i = 1, \dots \dots, n$
And $N(x,y)$ is taken to be a Gaussian symmetric probability density function

Finding the buildings

The above probability density function only takes into account the special coordinates of each of the detected points. This is not enough information to detect a building reliably, therefore this information must be combined with the gradient direction and magnitude as mentioned previously.

To do this we combine the spatial coordinates, (x, y) with the gradient direction, θ and magnitude, ω to establish the effect of each corner point i on the building probability. This is done by shifting each corner feature by θ, with half the weight to approximate the building center as shown below:

$$\begin{aligned} \hat{x}_i &= x_i + 0.5\omega_i \sin(\theta_i) \\ \hat{y}_i &= y_i + 0.5\omega_i \cos(\theta_i) \end{aligned} \tag{10}$$

Now that the effect of the gradient weight, and direction have been taken into account, the new estimated probability density function is shown:

$$p(x, y) = \frac{1}{R} \sum_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma_i}} exp\left(-\frac{(x - \hat{x}_i)^2 + (y - \hat{y}_i)^2}{2\sigma_i}\right) \tag{11}$$

Where σ_i is the variable scale parameter for each detected point $i = 1, \dots \dots, n$ based upon the weighting of the corresponding point.
And R, represents the normalising constant of the probability density function.

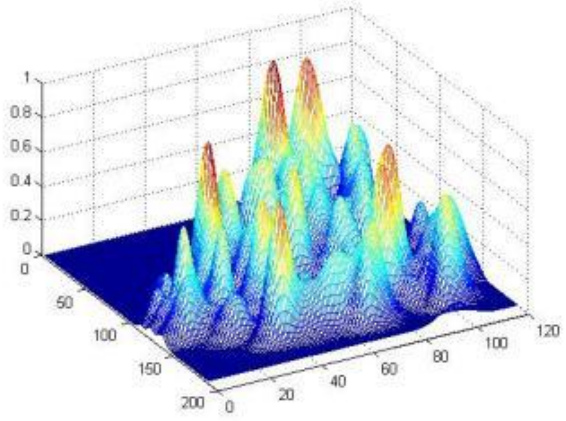


Figure 4-3 Typical building detection PDF with multiple peaks

Now that the probability density function has been established, it can be noted that due to the unknown number of buildings in a given image, the resulting probability density function for the building detection will be multimodal, where each peak represents a possible building center. Each peak does not always represent a building center, therefore a threshold is applied to ensure that only the highest probability values are used for building centers. This values is obtained by finding the peak with the highest probability value. This point is the most likely point to be a building center. Based on this maximum probability values, all peaks that are not at least $0.4 \times p_{max}(x_{max}, y_{max})$ are disregarded.

Now that the peaks of the probability density function have been threshold the final step is to extract the building centers. The peaks of the probability density function still no not represent building centers, and for this reason some morphological operations are done around the peak points (dilation by 6 pixels) to group points together that are close to one another, the centroids of the connected regions are then extracted to approximate the building center. It is important to note that this value will be sensitive to altitude of the aerial images, as at high altitudes many different buildings could be merged into one using this method.

4.2 Experiments and Results

A scenario of a level flight at an altitude of about 600m (~2000 feet) above ground level is assumed. Therefore a variety of aerial images where obtained from Google earth at this altitude for testing. Additional altitudes will be tested as many of the parameters need to be varied with altitude; however, the initial focus is to obtain reliable building detection results at a fixed altitude.

The results are presented in the form of true detection (building correctly detected) and false detection (building incorrectly detected). A building is assumed to be correctly detected if the detected feature point falls onto the building.

4.2.1 Feature Detector parameters

The first test was to tune the different feature detectors for building detection. A series of aerial images where assessed and the results are shown in this section.

Harris Corner Detector

κ is a tunable parameter used in the computation of the response function for the Harris corner detector. It is recommended by the authors that this value should be between 0.04 - 0.15 [19] for most applications. Therefore for test images, the parameters where tested from 0.04 to 0.15 in steps of 0.03. The results are shown below.

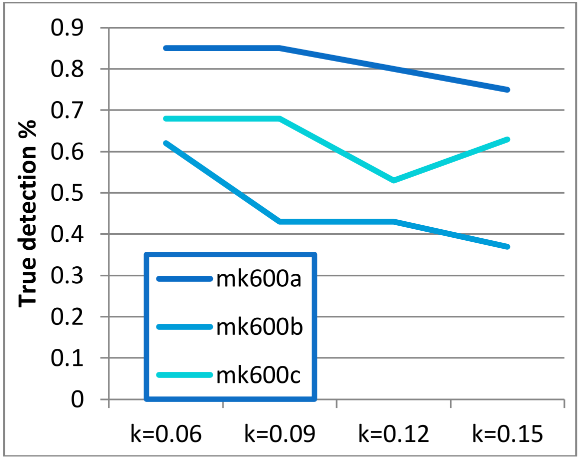


Figure 4-4 effect of κ on building detection

The best building detection rates are found with setting κ at a fixed value of 0.06. The higher values of kappa demand stronger corners to be detected, resulting in fewer feature points. Although this results in less false detection, the overall effect reduces the amount of true buildings detected.

FAST Corner Detector

The FAST corner detector is claimed to perform faster than the Harris corner detector [20] and therefore is a good candidate for corner detection. The only tunable parameter is the threshold which dictates the quality of the corners to be found. The images below show the effect of the threshold on detected corners.

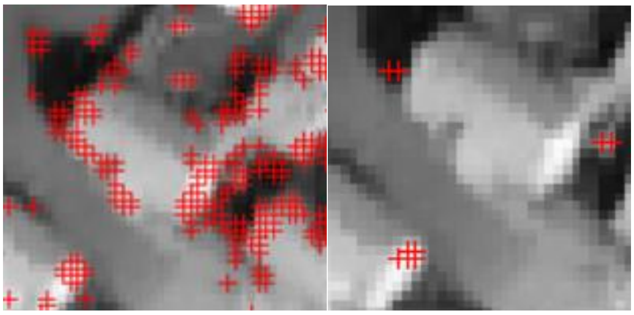


Figure 4-5 FAST corner features with threshold of 20(left), and 60(right)

The threshold value, T was tested between 20 – 60 and the results are shown below for the test images.

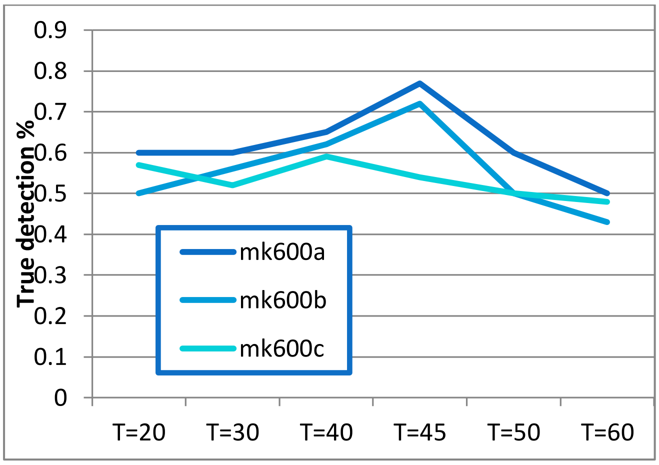


Figure 4-6 Effect of FAST threshold on building detection

The best building detection rates are found around a threshold of about 45. It is also noted that setting the threshold too low results in a large amount of feature points which increase the computational load considerably. The higher threshold value results in only strong corners which means only a few buildings are detected, although there are very few false detections.

Garbor Filter

The number of orientations is one of the parameters that can be adjusted in the Garbor filter.

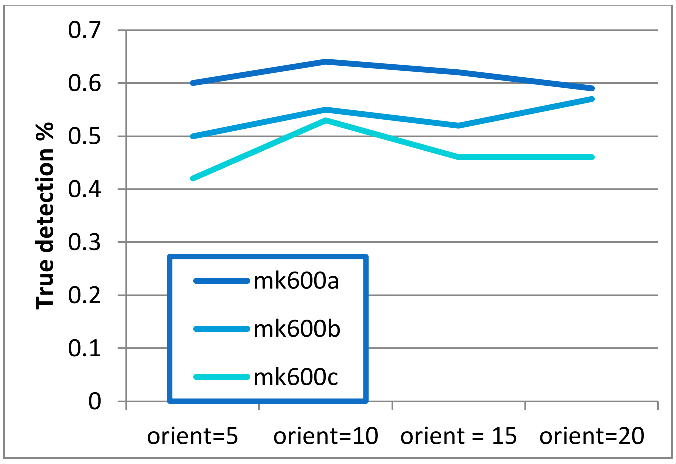


Figure 4-7 Effect of number of orientations on building detection

It can be seen that the orientation does not have a large effect on the amount of buildings detected, and therefore a value of approximately 10 was chosen as it gave the best results.

Another parameter that can be changed in the Garbor filter, is the frequency *f*. This value was tested between 0.55 and 0.85, the results are shown below.

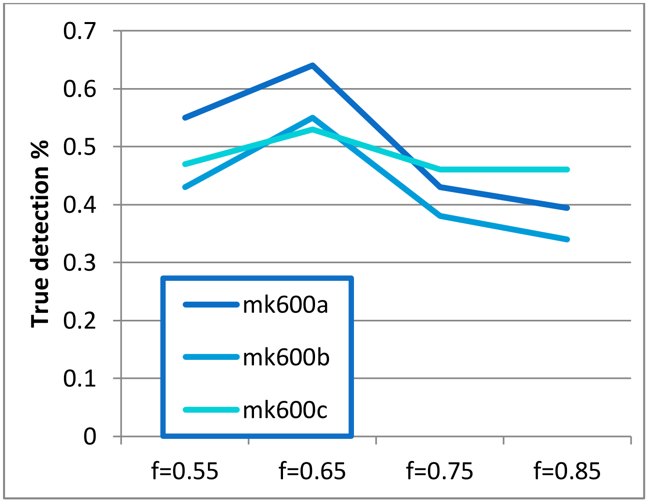


Figure 4-8 Effect of Garbor frequency on building detection

The frequency has a significant effect at higher values on the amount of true detected buildings. Based on these results the frequency will be set to be 0.65.

4.2.2 Building Detection Results – 600m

After the parameters for the corner detectors have been adjusted, a test can be performed to assess the effectiveness of the building detection algorithm. The initial test will assume the UAV is in a cruise altitude of 600m above ground level. The initial test will assume the altitude is constant. A variety of 11 test images were taken to simulate some of the areas the UAV will fly over around the Milton Keynes area. The spatial resolution is approximately 0.4m per pixel. Once again the 3 different feature detectors are used for corner detection.

The averaged building detection results are shown in the table below, for all 11 test images. In the 11 test images there are a total of 456 buildings which were manually counted. The tests were conducted using an Intel core 2 duo 2.00 GHz processor in the Matlab programming environment.

Table 1 - Performance of different feature detectors

Detector	True Detection %	False Detection %	Accuracy Ratio
Harris Corner	66.50	43.91	1.23
FAST	76.82	37.75	2.27
Garbor Filter	82.61	22.38	3.53

The processed images with the building detection points are shown and discussed in the following sections. Please note that for the purpose of clarity the images below are sometimes cropped from the full image.

Harris Corner Detector

A few of results for the Harris corner based building detection are shown below.

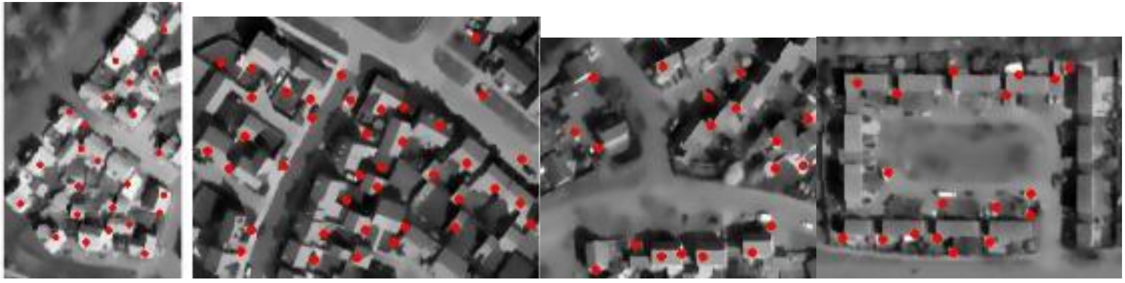


Figure 4-9 Building Detection Results using Harris Corners

Generally the Harris corner based building detection performs poorly. The main problem is a large amount of false building detection points. Like the other methods, the false building detection points are often scattered around actual building locations.

The speed of the Harris corner point based feature detector is amongst the fastest at an average computational time for a 640x480 pixel resolution image about 2.83 seconds in the matlab environment.

Table 2 - Speed of Harris Corner based building detection (matlab with 2GHz intel core 2)

Process	Time (sec)
Image smoothing	1.23
Feature extraction	0.83
Probabilistic search	0.77
Total	**2.83**

FAST Corner Detector

The results for FAST based building detection are shown below

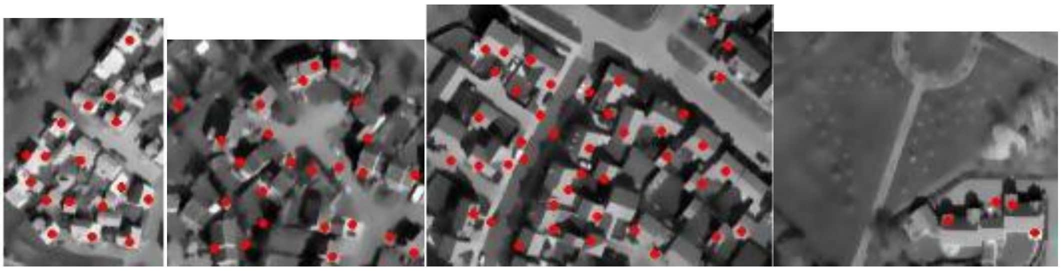


Figure 4-10 Building Detection results using FAST corners

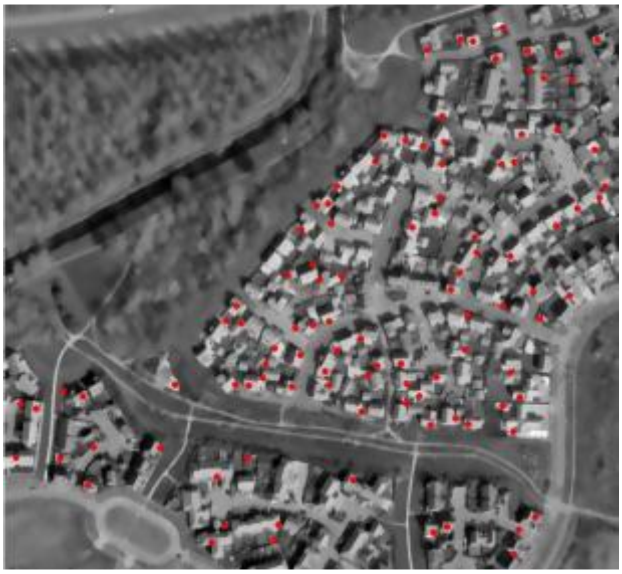


Figure 4-11 Building Detection results using FAST corners

The FAST feature detector performs better in terms of true building detection; however, there is still a large amount of false building detection rates for some images. The false detections are mostly located very close to building positions so more work should be carried out to try and reduce these poor points. The false detection points are often caused by building shadows, or square shaped objects that are not necessarily a building such as a parking lot or car. The main reason for buildings that are not detected is due to the colour that is similar to that of the road/ area around it. This results in weak corners that have little weighting

The speed of the FAST corner point based feature detector had an average computational time for a 640x480 pixel resolution image of about 5.51 seconds in the matlab environment. The probabilistic search takes longer than the Harris corner due to the fact that many more feature points are extracted in the image which results in more points that need to be processed. Although the authors claimed that the FAST corner detector was faster than the Harris corner detector the matlab implementation does not share this characteristic due to the nature of matlab being an interpreted instead of a compiled language.

Table 3 - Speed of FAST Corner based building detection (matlab with 2GHz intel core 2)

Process	Time (sec)
Image smoothing	1.23
Feature extraction	1.12
Probabilistic search	2.16
Total	**4.51**

Garbor Filter Detector

The results for the Garbor Filter based building detection is shown below

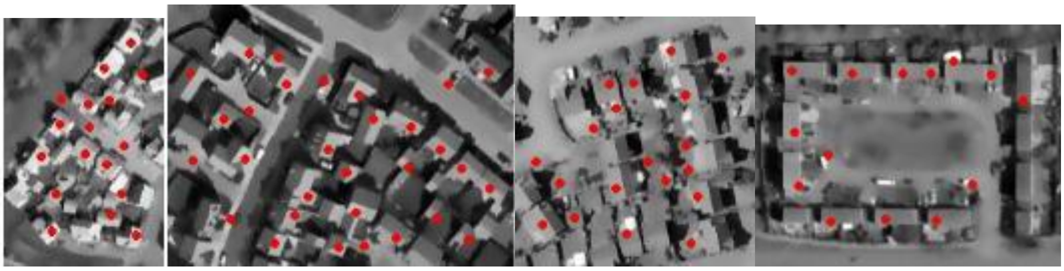


Figure 4-12 Building Detection results using Garbor Filters

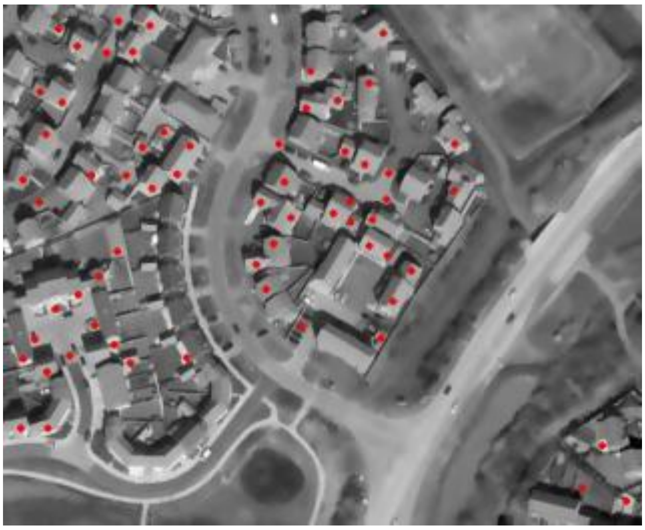


Figure 4-13 Building Detection results using Garbor Filters

The Garbor Filter based building detection method has the fewest false positives when compared to the other two methods. There are still a few false building detection results, but once again they are very close to building locations. Also there are some buildings that are detected more than once which could potentially cause problems with the geolocation system.

The speed of the Garbor Filter point based feature detector has an average computational time for a 640x480 pixel resolution image of about 6.37 seconds in the matlab environment. This the slowest method for feature detection, however the most accurate results are obtained.

Table 4 - Speed of Garbor Filter based building detection (matlab with 2GHz intel core 2)

Process	Time (sec)
Image smoothing	1.23
Feature extraction	1.83
Probabilistic search	3.31
Total	**6.37**

Altitude sensitivity

A brief test was performed at different altitudes to test how performance changes. The general conclusion is that the amount of false detection points greatly reduces as altitude increases because there is less noise in the image as small objects are not visible.

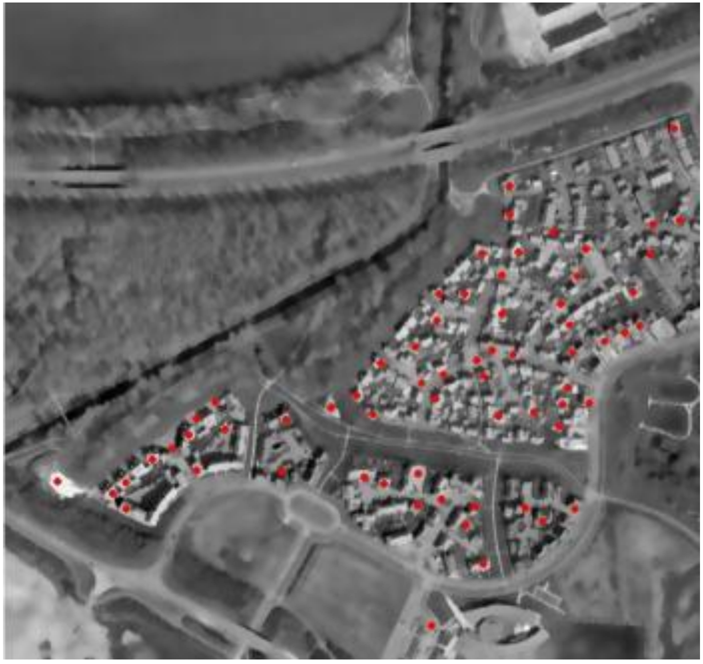


Figure 4-14 Building detection at 1000m

The image above shows very poor true building performance. This is mainly due to the averaging function in the algorithm that extracts the peaks of the probability density function. This averaging process blends points within the radius of the peak, and therefore for this image blends many different buildings together. This value will need to vary linearly as altitude changes to avoid merging building points.

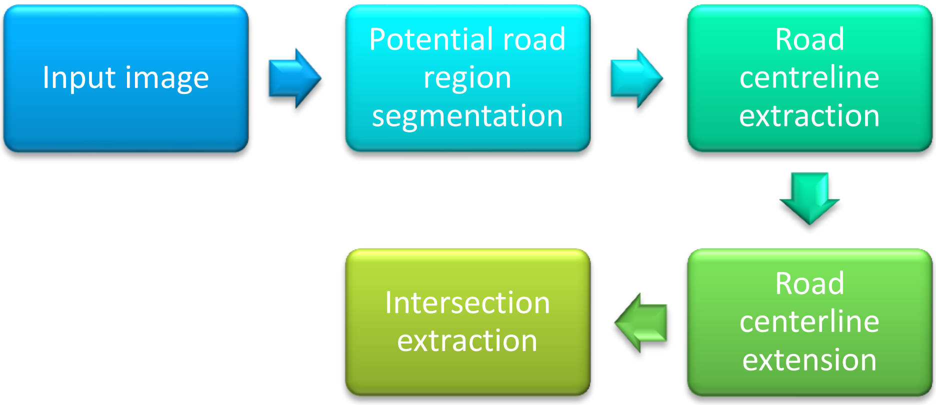
Edge based building detection Summary

Overall it can be seen buildings can be found in a given aerial image using corner/edge features. Three main corner features were tested and it was found that although the Garbor Filter method produced the most accurate results, it was also one of the slowest methods. This is mainly due to the convolution operations which are computationally expensive. In general all the methods are fairly slow and are not suitable for real time applications onboard a UAV. The FAST based detector seemed to perform best in terms of speed and accuracy

Most of the work carried out in aerial image object detection is used in geographic applications such as GIS systems. In such applications speed and computational efficiency are not the driving factors so most algorithms used can take minutes or hours to process a single image, authors of [7] reported that their method took several minutes to process a 800x600 resolution image using matlab on a core 2 intel computer. The methods developed in this chapter are significantly faster, but are still not fast enough for use onboard a UAV. Therefore the methods used in this section are best suited for applications that do not require high speed, like geography based such as mapping, GIS systems, or planning.

5 Line Based Road Detection

Roads in aerial images can be considered to be curvilinear features. Based on this property it is appropriate to adopt the curvilinear structure detector that was proposed by Carsten Steger [3]. It can be noted that this detector can also be applied to detection of other linear features in aerial images such as rivers, railways, and hedge lines, which will be useful in remote areas where no roads or buildings can be detected. However, the main focus of this section is the detection of roads. The road extraction algorithm is described below


Input image
Potential road region segmentation
Road centreline extraction
Road centerline extension
Intersection extraction

The first step is to segment out the potential road regions in order to remove any noisy regions that are caused by buildings, vegetation. This is done by simple colour segmentation that has fixed values based on extensive testing. After the potential road regions have been extracted the next step is to extract the curvilinear features using a method proposed by Carsten Steger [3]. At each step it is important to perform some morphological operations to smooth and filter the regions. Once the main road centre lines are extracted it is necessary to extend the endpoints as the centre lines are not detected near intersections due to the fact that roads generally spread out at intersection points. Once the lines have been extracted the last step it to extract the intersection locations.

5.1.1 Road Segmentation

To help with extracting the road centre lines the image is segmented into regions that match the colour of roads. The majority of roads can be considered to be asphalt, however there are some rural regions where there is a large number of dirt/gravel roads. Therefore the image will be segmented into regions that match dirt roads, and regions that match asphalt roads based on colour information. As this project is aimed at developing a geolocation system primarily for use in the UK, only roads in the UK will be used.

Asphalt roads are typically dark and black/grey, however the colour does vary due to the different age of roads, or shadows. Below are some images illustrating the variation of asphalt roads.

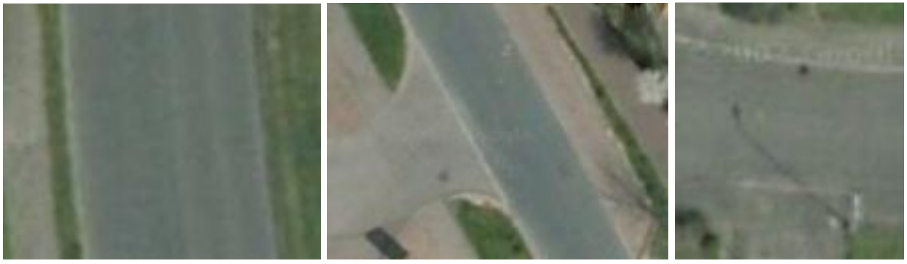


Figure 5-1 – Various asphalt roads extracted from aerial images

Similarly there is also a variation in dirt/gravel roads as shown below, however dirt roads are generally a lighter colour.

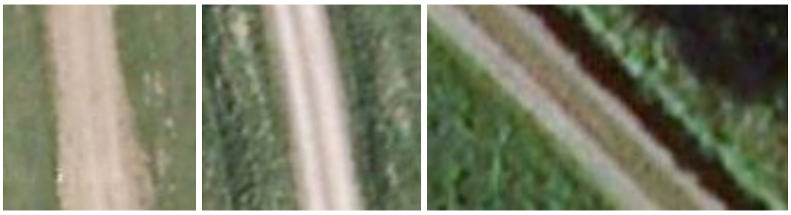


Figure 5-2 - Dirt/Gravel Roads extracted from aerial images

Based on a random sample of aerial images from around the UK, suitable threshold values were chosen that allow the road regions to reliably be extracted from aerial images to aid the road centre line extractor.

Various different colour spaces were considered when attempting to segment out the road regions. These included RGB, YCrCb, HSV and L*a*b* colour space. Below Figure 5-3, Figure 5-3 and Figure 5-5 illustrate the YCrCb and Lab colour spaces. Figure 5-5 shows the HSV colour space for the same image taken of a typical suburban area.

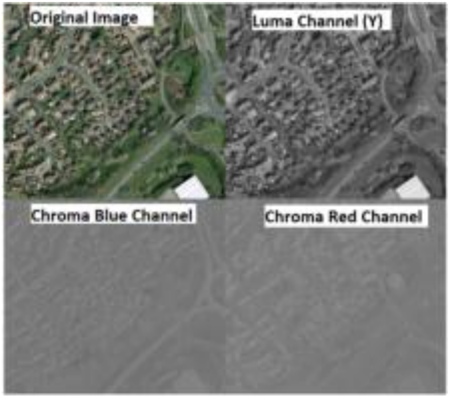


Figure 5-3 - Separated Y, Cr, Cb channels

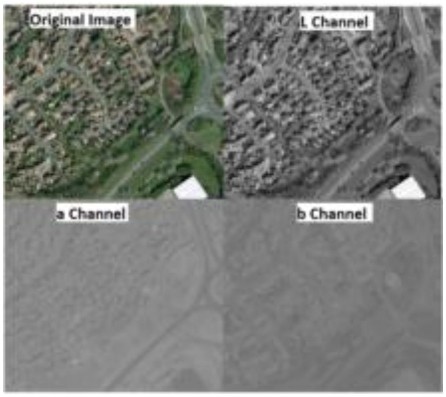


Figure 5-4 - Separated L*, a*, b* channels

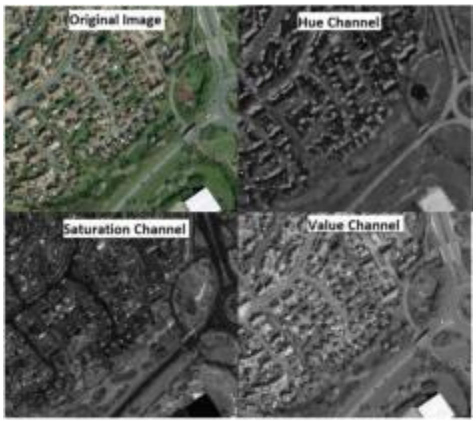


Figure 5-5 - Separated H,S,V Channels

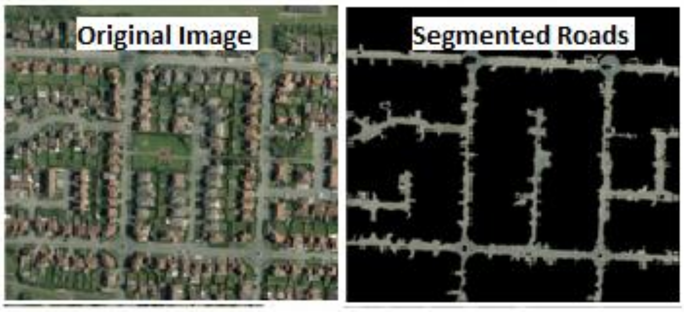


Figure 5-6 Segmented road regions

It was found that the most suitable colour space to segment out the road regions is a combination of HSV colour space to make use of the general colour of the roads, with the added saturation channel from the HSV colour space to aid accuracy in road region extraction. Although the L*a*b* colour space is known to be useful for colour based segmentation, adequate results were found using the HSV colour space with the added advantage of computation efficiency compared to L*a*b* colour space.

It can be seen that the roads are most obvious in the saturation channel for many different scenes. The HSV colour space allows for the dark grey/black region to be found easily, and the other colour buildings and vegetation to generally be rejected. The saturation channel allows the road regions to be found accurately while rejecting some of the darker buildings. There is generally some difficulty distinguishing dark roof buildings that are near roads, but the combination of the 4 channels, and some morphological operations such as opening and closing reduce some of the noise.

The HSV threshold values are not very tight; this is to mainly remove unwanted vegetation areas and allow the segmentation to work on a variety of different scenes of varying light. For the saturation channel, the thresh holding is performed dynamically based on otsu's automatic threshold selection.

5.1.2 Road Centre line Extraction

In order to extract the road centre lines the curvilinear feature detector proposed by Stegner in [3] is adopted. This algorithm was chosen because it was primarily developed as a low level computer vision detector for remote sensing tasks such as road detection; however it has other uses in medical imaging. The curvilinear feature detector returns sub pixel line detection accuracy as well as the line width which is particularly useful for road detection.

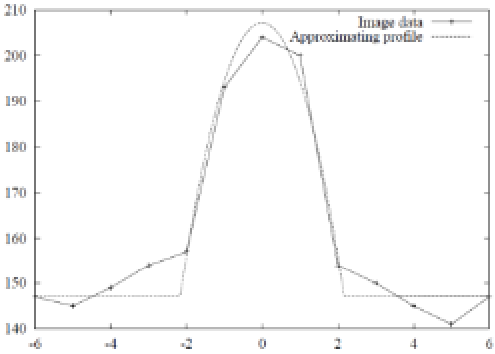


Figure 5-7 Typical gradient profile of a line

A curvilinear feature (in this case a road) $z(x)$ can be modeled as a curving line where $p^{\rightarrow}$ is the direction along the road and $n^{\rightarrow}$ is perpendicular across the road at any point. A typical profile across a road in terms of grey scale value in the direction $n^{\rightarrow}$ takes on a parabolic or rounded profile as the intensity values change across the road. The road centre line can be found by taking the derivatives in the direction $n^{\rightarrow}$ where the first order derivative of the road is zero and the second derivative reaches a maximum value. In real situations there is noise present and therefore the image is convolved by the derivatives of Gaussian smoothing kernels

The local direction of the line can be found using partial derivatives of the image r_x, r_y , r_{xx} , r_{xy} , r_{yy} . These are estimated by convolving the image with Gaussian smoothing kernels. The Gaussian smoothing parameter σ should satisfy the condition of:

$$\sigma \geq \frac{w}{\sqrt{3}}\ for\ a\ road\ of\ width\ 2w \tag{12}$$

The road direction is found by computing the eigenvalues and eigenvectors of the Hessian matrix for each pixel:

$$H(i,j) = \begin{bmatrix} r_{xx(i,j)} & r_{xy(i,j)} \\ r_{xy(i,j)} & r_{yy(i,j)} \end{bmatrix} \tag{13}$$

The likelihood of a centre line is represented by the maximum absolute value of the eigenvalues λ, and the corresponding eigenvector is the direction across the road, $p^{\rightarrow}$
To find the sub pixel location of the centre line at the point where the first order derivative of the road in (n_x, n_y) is zero. This point can be calculated as:

$$\begin{aligned} (p_x, p_y) &= (tn_x\,, tn_y) \\ where\ t &= -\frac{r_x n_x + r_y n_y}{r_{xx} n_x^2 + 2r_{xy} n_x n_y + r_{yy} n_y^2} \end{aligned} \tag{14}$$

If this point satisfies $(p_x, p_y) \in \left[-\frac{1}{2}, \frac{1}{2}\right] \times \left[-\frac{1}{2}, \frac{1}{2}\right]$ it can be considered to be a point on a line.

Once all of the line points have been found, it is required to link them into a line. To do this the point with the highest second derivative it chosen as this is the most likely point to be a road centre line point. For each of the detected points the lines will be constructed by adding the appropriate neighbor in the direction of $p^{\rightarrow} = n^{\rightarrow} \pm \frac{\pi}{2}$. The choice of the appropriate neighbor to add is based on the sum of the distance between the respective centre line points, and the angle difference as shown in the choice function:

$$\begin{gathered} S = \left\|\left(p_x^2, p_y^2\right) - \left(p_x^1, p_y^1\right)\right\| + |\alpha^2 = \alpha^1| \\ superscript\ 1 = \ current\ centreline\ point, superscript\ 2 = \ previous\ centreline\ point \\ \alpha = angle\ of n^{\rightarrow} \end{gathered} \tag{15}$$

This linking process will continue until all the current neighbor pixels have a maximum eigenvalue λ below a threshold.

Due to many buildings, or house driveways having a similar colour to the road, the road segmentation step results in many building regions being included, this sometimes causes additional lines to be detected. To overcome this issue, a post processing step is introduced. Morphological operations such as thinning are performed on the road centre line image to remove some noisy regions. An example of the before and after the post processing step is shown below.

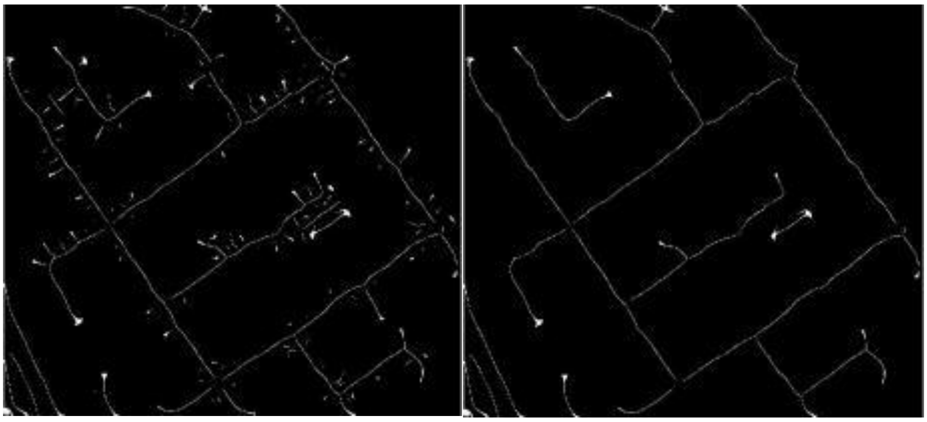


Figure 5-8 Road extraction before (left) and after (right) post processing

5.1.3 Road centreline extension

The road centre line detection algorithm generally does not work well at the location of junctions as the width of a road increases at junction areas. This causes problems in the road centre line extraction and most junctions are not found. However the roads are detected up to the point of the junction and therefore the detected road centre lines are extended 3w (w=width) along the direction of p→. If the extended centre line intersects with another it is marked as a junction.

5.1.4 Road Intersection Detection

Once road endpoints have been extended, a hit and miss transform[23] is applied to find the intersections. The hit and miss transform searches a binary image for regions that match the structuring element. In this case the structuring element used is a 3 way intersection, and a 4 way intersection.

5.1.5 Experiments and Results

Before testing of the entire geolocation system, it is first important to obtain reliable road detection results. Once the road regions are detected accurately the next step will be intersection extraction. This section will detail some tests and results for road detection from aerial images around England.

A scenario of a level flight at an altitude of about 600m (~2000 feet) above ground level is assumed. Therefore a variety of aerial images were obtained from Google earth at this altitude for testing. A test image with the segmented road regions, and road centre lines is shown.

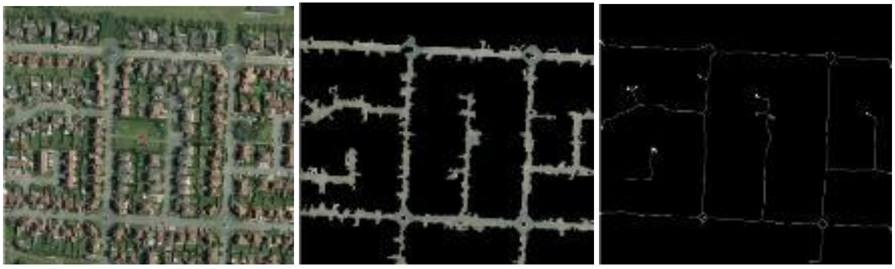


Figure 5-9 Road detection results, original image (right), road regions (centre), road centerlines (left)

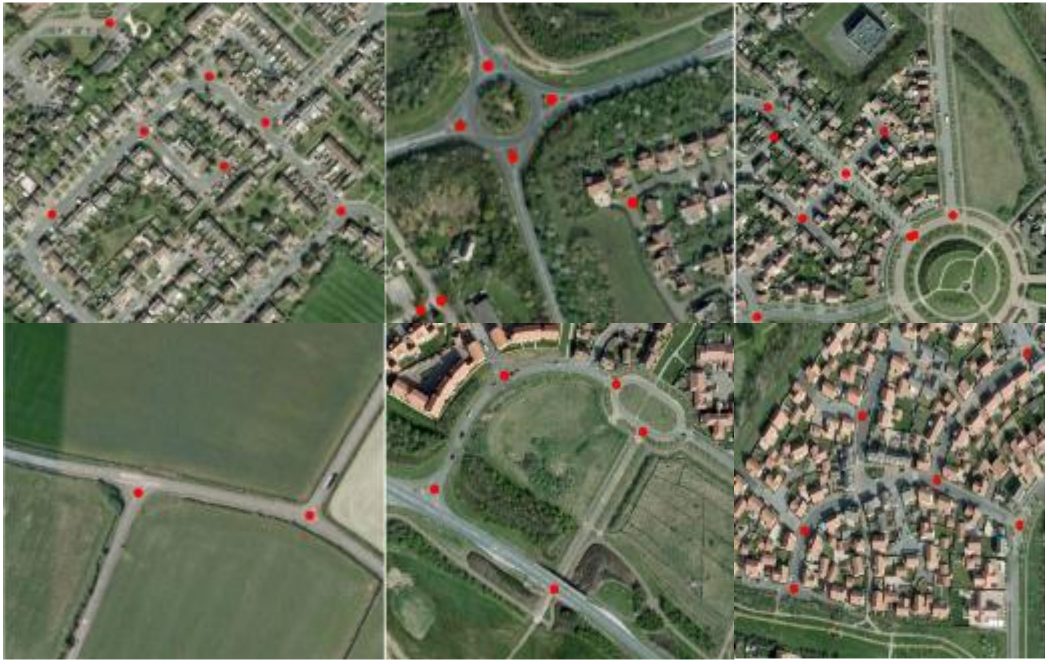


Figure 5-10 Selection of road intersection detection results from aerial images

As shown in the images above, the road intersections are often accurately extracted with very few false positives. Of the 12 test images there was a total of 53 intersections which were manually counted. It was assumed that only roads count as intersections, small road areas such as driveways do not count as an intersection.

Table 5 - Performance of Road Intersection Detection

No. intersections	True Detection %	False Detection %
53	62.7%	19.2%

The averaged results of the road detection tests are shown above. Generally the true detection points are very accurately located in the centre of the intersection which is ideal for the geolocation system. A few false detection points were found, this is generally due to building roofs that have a similar colour to the road close to it. Another particular problem with false detections arises when one road crosses over another, this is not necessarily an intersection as there is a bridge.

Altitude sensitivity

A brief test was performed for different altitudes up to 1000m above ground level to test how performance changes.

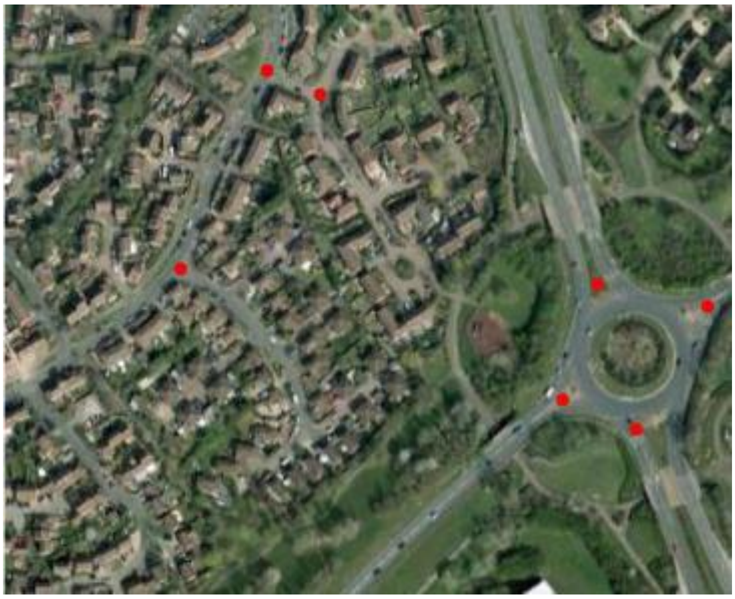


Figure 5-11 Intersection detection at 1000m

The general conclusion is that for a constant window width, acceptable performance is achieved for slight variations in altitude (~200m) around the baseline of 600m, at this scale the width of the roads do not change considerably. It was found that at lower altitudes, the curvilinear feature detected regions such as building edges as roads so the performance at low altitude was poor. However at much higher altitudes superior performance was achieved do to the roads standing ot more clearly. The window width parameter of the road lines to be detected needs to vary otherwise narrow roads, or very wide roads will be missed.

Line Based Road Detection Summary

This chapter has shown that it is possible to detect road intersections and roads in an aerial image. As with the work conducted in Chapter 3, the line based road detector is based on curvilinear features in an image, which is less sensitive to seasonal and weather changes as roads will always represent fairly long linear regions. Although the colour of an asphalt road remains fairly constant under different conditions, the fact that the filtering relies on colour information limits the overall envelope of this detector, particularly in a situation of snow. Furthermore the high computational burden of this method once again means that this technique is not suited for UAV geolocation where computational capacity is limited.   This method for detecting roads will be more suited for geography based applications where a user can monitor the conditions and adjust the parameters when the environments change.

6 Object Detection using Haar classifiers

The previous methods of building/ road detection have been shown to work in controlled tests once the parameters are tuned for specific conditions. When the conditions vary such as lighting, or colour, the detection results break down. This section introduces another method for object detection that should be more robust to changing conditions. Instead of relying on image pixel intensities (Red, Blue, Green pixel values) which are computationally expensive, this Haar classifier based approach makes use of Haar-like features. The main advantage of Haar-like features is the high speed of feature calculation. This should overcome the speed issues of the previous methods, enabling near real-time landmark detection to be used onboard the UAV.

Haar-like feature detection looks at rectangular regions in an image and calculates the sum of pixel intensities, and finds the differences between each of these regions. Computational efficiency is achieved through use of Integral Images as discussed in chapter 0 which allow for very fast summations in the image. Once the integral image has been calculated a Haar-like feature can be found at any scale at any pixel location in a few operations. An example of haar like features for a face is shown in Figure 6-1.

On a face, the eye regions are typically darker than the cheek or nose regions. A Haar-like feature will approximate the light and dark regions as shown in Figure 6-1

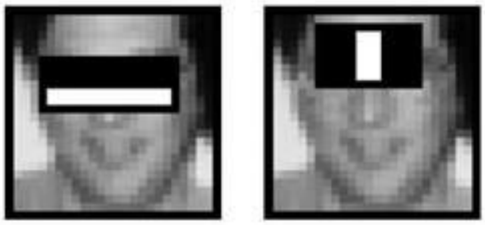


Figure 6-1 Example Haar-like features of a face

Originally a set of basic Haar like features used in[11] which consisted of perpendicular edge and line features. Recently Lienhart et al improved the performance of the haar cascade by introducing addition haar like features at 45^{o}[24]. An example of the Haar like features used by Lienhart is shown below, these are known as the Extended Haar like features.

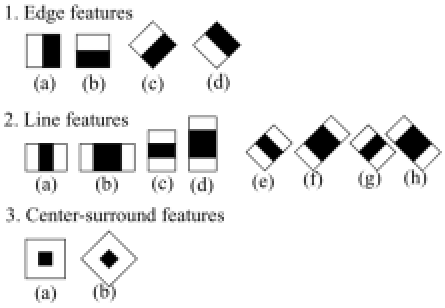


Figure 6-2 Haar-like feature set

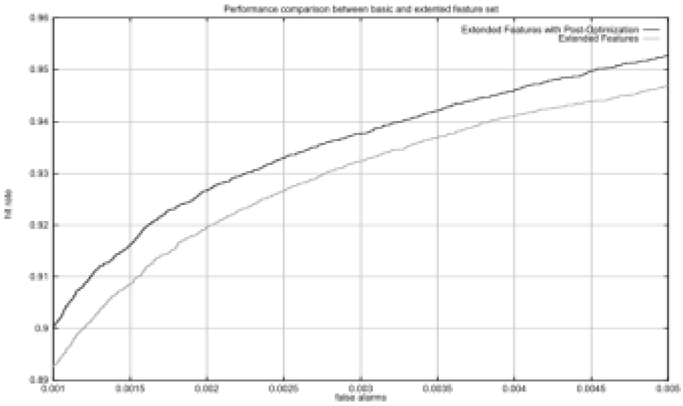


Figure 6-3 ROC curve for both Haar feature sets and the extended feature set

Tests conducted by Lienhart showed that there was a performance increase when using the extended Haar like features. As shown in the ROC curve in Figure 6-3, the extended set of features reduced the false alarm rate by about 10%. Due to the performance increase, the extended Haar like features will be used in this thesis.

6.1 Haar Classifier Cascade

A Haar Classifier consists of many weak classifiers that form a strong classifier. The haar features are taken from the response of the Haar functions (Figure 6-4) within the image at a given orientation. This way a collection of weak classifiers are used to create a strong classifier. Weak classifiers are not good enough to reliably detect any object, they are only required to be better than chance, and can be simple and computationally inexpensive. When the weak classifiers are combined as a cascade, they form a single strong classifier as shown below.

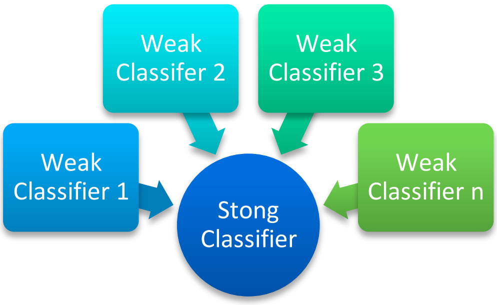


Figure 6-4 Haar strong classifier node structure

The strong classifier is capable of detecting a common structure at different illumination, colour and scale. These features make the Haar classifier ideal for use in aerial image detection, as aerial images are subject to different illumination and colour at different times of day/weather conditions. Also being able to detect objects at different scale is critical for a UAV that will fly at different altitudes.

A single strong classifier is not sufficient for reliably detecting objects as there is still a chance of false positive detections. In order to improve the performance, multiple strong classifiers are combined again to from a Haar cascade classifier. Machine learning techniques are used to create the Haar cascade classifier and are discussed later in Chapter 6.2. The haar cascade classifier consist of many strong classifier nodes to form a degenerative decision tree with haar like features. Each region of the image is passed through the cascade, the given region will need to achieve a pass response from all of the classifiers in the cascade to be successfully classified. For performance improvements, the classifier is sorted in order of most discriminative feature during training. This means that the haar classifier cascade combines successively more complex classifiers which eliminate negative regions quickly, while spending more time on more promising regions in the image.

Dealing with Rotation

Although the haar classifier is able to handle changes in scale, illumination and colour, it is not able to deal well with orientation. A basic test was conducted to evaluate the ranges of orientation that the haar classifier is able to deal with.

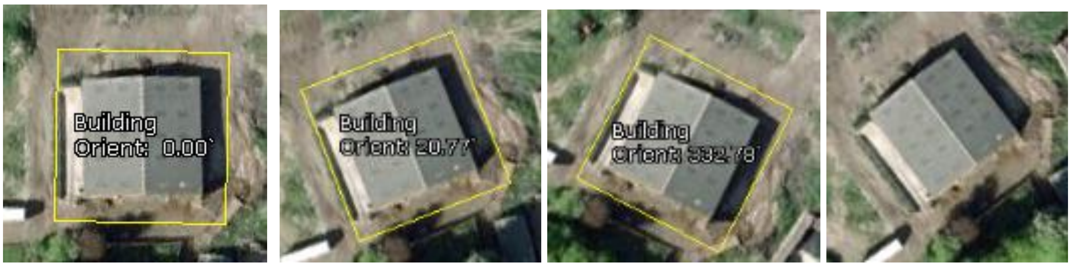


Figure 6-5 Orientation sensitivity of a building Haar classifier

In this test a simple single strong haar classifier was trained and tested on the same image. The same roof image was then evaluated using the classifier, when the image was not rotated; the classifier correctly detected the roof correctly. As the image was rotated the haar classifier was still able to detect the roof up until orientations of about 20^{o} to 25^{o}.

As buildings in aerial images are not going to be fixed in orientation, buildings at different orientations will not be detected. There are two possible solutions to this problem. The first is to train multiple classifiers at each major orientation (0^{o}, 45^{o}, 90^{o}, 135^{o}). This approach will require a huge training dataset to deal with the variations in builings and orientations.

The alternative approach that is used in this thesis is to use a single classifier, and rotate the source image. This allows for a simplified dataset and faster training. The image is evaluated by the single classifier at different orientations. The additional processing to rotate an image through different rotations is minimal. The source image will be evaluated at each major orientation in steps of 45^{o}.

6.2 Haar Training

This section describes how the Haar classifiers were trained to detecting buildings and road intersections in aerial images.

6.2.1 Boosting

The Haar Classifier Cascade relies on machine learning techniques to train the classifier, many different machine learning techniques can be used, however the most common method is to use boosting.

Boosting is a powerful supervised learning technique that is used to construct the Haar Cascades. A supervised learning technique requires a dataset of labeled positive images, and a dataset of negative images that do not contain the object to be detected. More information of how the training dataset as obtained is found in chapter 6.2.2. The main variations of boosting are known as Discrete Adaboost, Real AdaBoost, and Gentle AdaBoost. All the variations construct a cascade of classifiers that are comparative in terms of computational efficiency for object detection; however the learning techniques differ from each boosting algorithm. Each stage of the cascade is trained to detect almost all of the objects of interest, while rejecting a certain fraction of the non-object regions.

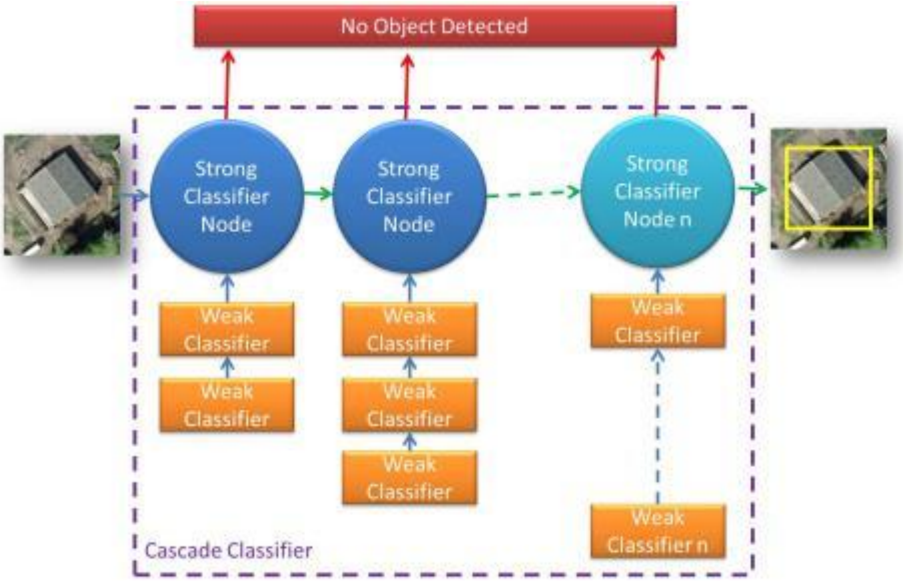


Figure 6-6 Haar cascade structure

The AdaBoost algorithm is used to select good features to use on each of the weak classifiers. Then using the large base of weak classifiers, a strong classifier is learnt. As mentioned before, weak classifiers on their own only perform better than chance, however when they are combined to form a strong classifier, the performance is improved greatly.

In this thesis the weak classifiers consist of a single haar feature, with a simple binary threshold decision. Each weak classifier is a stump in the decision tree, and the binary threshold is used to specify if a feature is present or not. These classifiers are added in a way that best classifies the weighted training sample. The AdaBoost algorithm then creates a strong classifier from each weak stump by adjusting the weights to find the optimum features which best separate the positive and negative training images. The weights are adjusted based on correct identification of a feature (increase weight) and incorrect detection or false positive (decrease weight). This will repeat until a specified true and false detection rate is achieved. As training progresses, a cascade of haar classifiers is constructed with the strong classifiers with the least weak classifiers at the beginning of the tree. This improves computation speed as only when a region has a good likelihood of being an object will it reach the later stages of the cascade. The regions that do not contain the object of interest will be removed quickly.

6.2.2 Training Dataset Creation

The Haar classifier is trained using a supervised learning technique which requires a dataset of labeled positive images (images with object to detect visible) and a dataset of labeled negative images (images with no object visible).

The two main landmarks that most suitable for geolocation are road intersections and buildings. It was also previously decided that a single orientation will be used to detect images at different orientations. In order to train the two classifiers for buildings, and road intersections a dataset for each object class is required. There are many publically available datasets that are provided by various research institutions, the most common being face, and pedestrian datasets. Because the problem of aerial image object detection has not been extensively studies a dataset was not found, and thus one was created.

Crowd sourcing dataset creation

Typical datasets require thousands of positive and negative training images. It would take a considerable amount of time for an individual to create a large dataset. In this thesis we made use of crowd sourcing technology to create the datasets. The crowd sourcing platform that was used was the Mechanical Turk[25] by Amazon. This platform allows users to submit simple Human Interaction Tasks (HIT), and other workers to complete these HITs. To encourage workers to complete the HITs a small reward will be paid to the worker for completing the HIT.

The Human Interaction Task for the dataset creation was to hand label buildings/ road intersections. To do this a large collection of aerial images were obtained from various locations across the UK. These aerial images were obtained from publicly available aerial image databases such as Google Earth, and Microsoft Bing Maps. The maps were obtained from various altitudes to help create a more diverse training set of different scales, the altitude was between 1000ft to about 3000ft. An example of some of the map tiles are shown below.

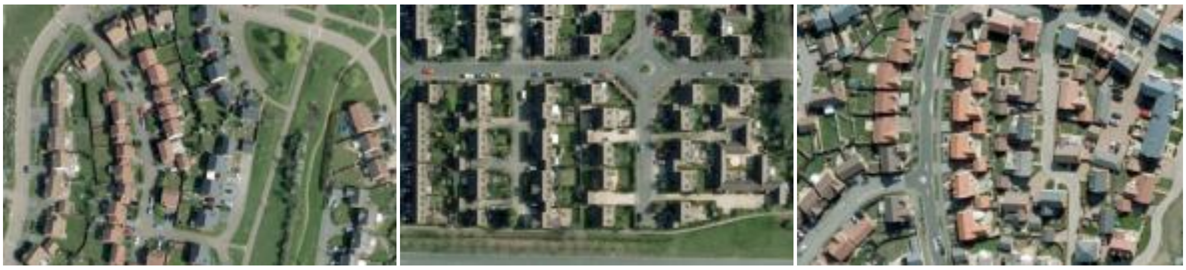
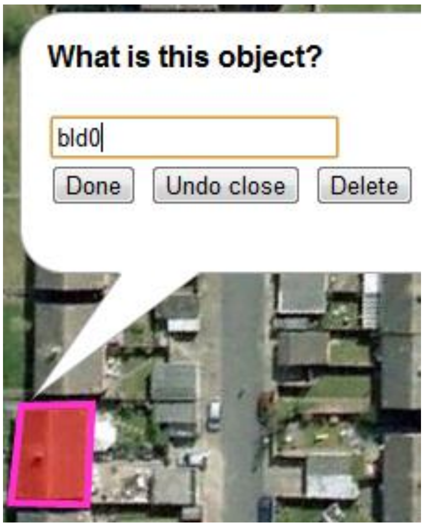


Figure 6-7 example map tiles used for object labeling

A total of 500 map tiles were obtained for the building labeling, and another 500 was obtained for the road intersection labeling. These images were then sent for processing on the Amazon Mechanical Turk. The Human interaction Task required the users to draw a polygon around each of the objects and give them a name, in this case the workers were also required to also specify the orientation of the object, as this is crucial for the haar training.

A simple interface was used that was based on the MIT Labelme toolset[26]. The labelme toolset is an online tool that allows users to easily label images using a simple point and click interface, based on its ease of use, open source framework, and online nature made it ideal for use on the Amazon Mechanical Turk service.



What is this object?

bld0

An example of the simple interface is shown to the left, the user simply clicks the points around the object they wish to label. Once the polygon has been closed/ completed a dialog wil pop up asking the user to give the object a name, in this case the user will label the object class (building/intersection) and specify the major orientation (0,45,90,135)/

The quality of the labeled objects was of concern, and for this reason each image had to be manually reviewed before it was accepted. It was noted that some users did not follow the instructions clearly but overall the few incorrect object labels could be manually corrected. An example of some good labels and poor region labels are shown below for buildings.

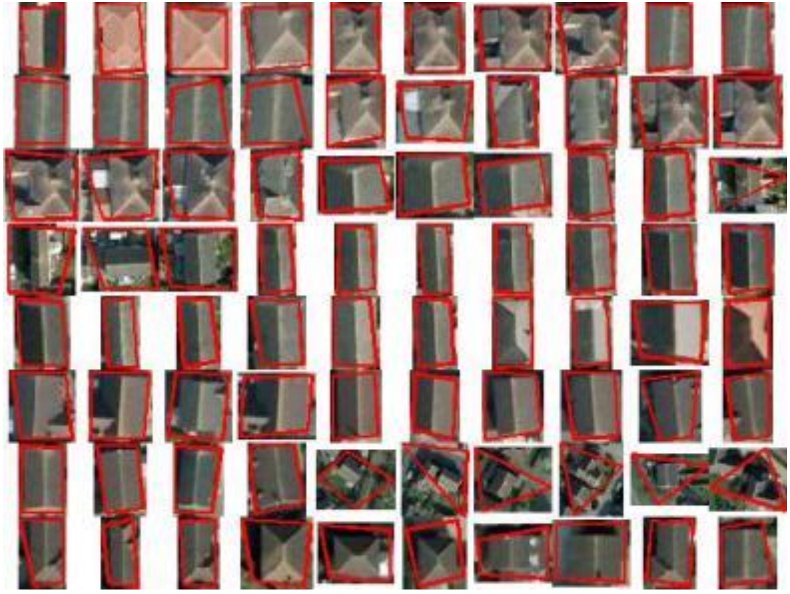


Figure 6-8 Building region labels from Crowd Sourcing

Approximately 5000 labeled objects were created for buildings and road intersections. The use of a crowdsourcing platform such as the Amazon Machanical Turk allowed for these images to be labeld in under 2 hours, whith an additional 2 hours of manual checking. This time is significaly less than the time it would take an individual to manually create these datasets which would take several days. The main disadvantage of this method for dataset creation is the fact that the quality of workers is not always knows so you will always need to manually review the labels, for our dataset we estimate that approximately 90% of the building labels were correctly identified and required no further editing. However the advantages are clear of using such techniques to create large datasets quickly.

Finally the negative training set can be extrected from the non object regions from the labeled images. Additionaly a large quantity of aerial images over rural areas were obtained that did not contain any road intersections or buildings. Theese images that did not contain and of the landmarks will be used as the negative dataset, which typically should be larger than the positive dataset. In this thesis a negate image dataset of approximately 15 000 images was used. An example of some images extracted from the dataset are shown below.

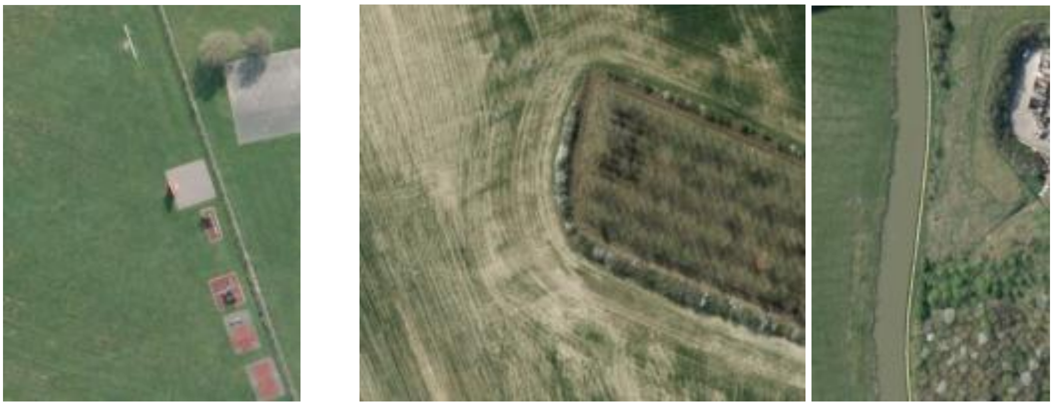


Figure 6-9 Example negative images used to train the haar classifier

Generally negative images can be any images as long as they do not contain the object you are trying to detect; however for best results you should use images that are in the same context (i.e. aerial images) to help reduce the number of false positive detections.

6.2.3 Training

As mentioned before, the Haar classifier is a supervised learning technique and requires labeled positive and negative images for training. A novel method for fast dataset creation was discussed in the previous section. This section will cover the training of the Haar cascade classifier.

The OpenCV computer vision library [27] includes a Haar Training utility that will train a Haar classifier from a given set of positive and negative images.

Creating Samples Vector

The haar training utility requires a special vector input of training images, so the first step is to create this .vec file. In order to generate the vector file, a text file with a list of positive images is required. A matlab script was created to this quickly and is described. The first step is to extract each of the square regions from the labeled object images. This rectangular region is taken as the extreme points of the labeled object polygon in x, and y. This is shown in the image below.

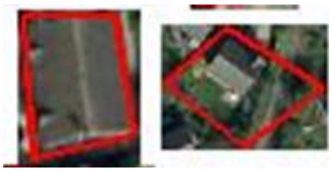
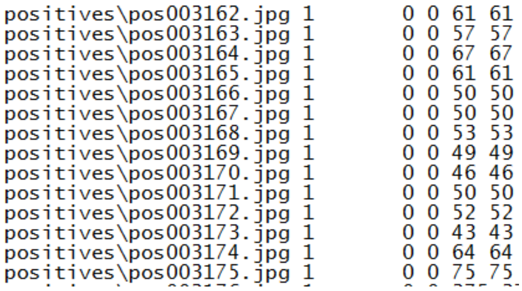


Figure 6-10 Square region extracted around polygon extreme points

Each of these square images, which are sub regions of the larger image are then stored on the computer, each as a separate image. The text (positives.txt) file is now generated which is essentialy a list of all the images that will be used for training. An example of this is shown below.



```
positives\pos003162.jpg 1       0 0 61 61
positives\pos003163.jpg 1       0 0 57 57
positives\pos003164.jpg 1       0 0 67 67
positives\pos003165.jpg 1       0 0 61 61
positives\pos003166.jpg 1       0 0 50 50
positives\pos003167.jpg 1       0 0 50 50
positives\pos003168.jpg 1       0 0 53 53
positives\pos003169.jpg 1       0 0 49 49
positives\pos003170.jpg 1       0 0 46 46
positives\pos003171.jpg 1       0 0 50 50
positives\pos003172.jpg 1       0 0 52 52
positives\pos003173.jpg 1       0 0 43 43
positives\pos003174.jpg 1       0 0 64 64
positives\pos003175.jpg 1       0 0 75 75
```

Figure 6-11 positives.txt for haar training

From left to right, the positives.txt file contains the relative image file location on the computer, the number of objects in the image, the local coordinates of the object (x1, y1, x2, y2). Because the images have already been cropped when they were extracted, the local coordinates of the object are simply the dimension of the image.

Similarly a negative images text file is required, as the negative images do not contain any objects the text file is simply a list of file locations of each of the negative images.

Once the text file has been generated, this can be passed into the sample vector application (opencv_createsamples.exe). The command line application requires you to call with the following arguments:

```
-info Positives.txt -vec PositivesMany.vec -num 5527 -w 14 -h 14
```

-info	positives text file which is a list of positive training images
-vec	name of vector file generated
-num	number of positive images
-w	width of positive samples vector
-h	height of positive samples vector
-show	show the sample vector once it is complete

The application will then proceed and create a .vec which will be used in the haar training application as discussed later in this section. The .vec file is essentially a list of positive images that have been specially formatted for the haar training application.

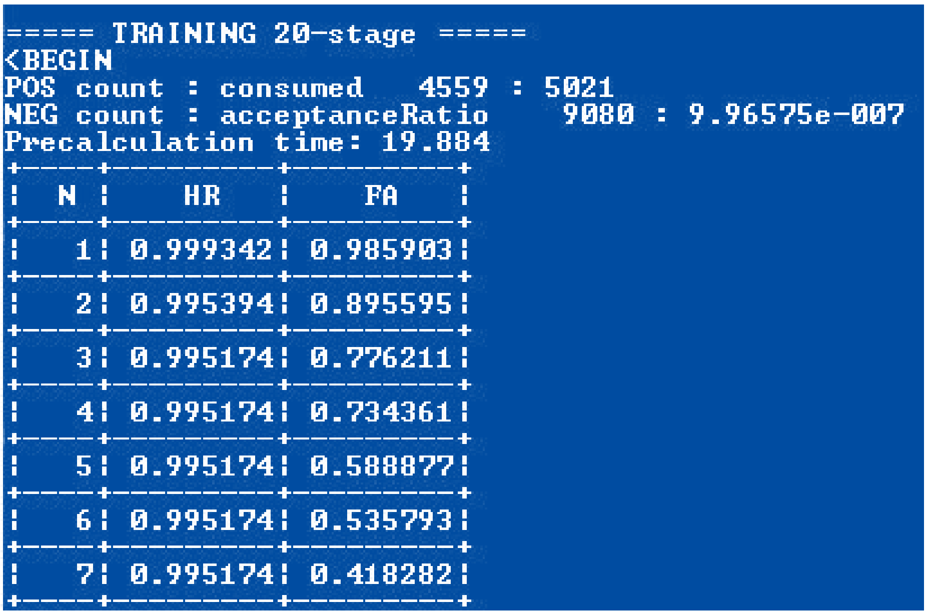
Training the classifier

Now that the .vec has been created it can now be fed into the haar training application (opencv_traincascade.exe). This application is called in a similar way as described before, however there are many more input arguments. An example of the arguments is shown below:

```
-data cascade -vec PositivesMany.vec -bg Negatives.txt
-numPos 5527 -numNeg 13000 -numStages  21
-featureType HAAR -w 16 -h 16 -bt GAB
-minHitRate 0.9950000047683716 -maxFalseAlarmRate 0.40
-maxDepth 4 -weightTrimRate 0.95 -maxWeakCount 100
-mode ALL
```

-data	relative path the the folder where the cascade will be generated in
-vec	training samples vector as shown previously
-bg	negative images list as text file
-numPos	number of positive training images to use
-numNeg	number of negative training images to use
-numStages	number of haar cascade stages
-featureType	type of features used for training, Haar features, or Local Binary Patterns(described in later section
-w	width of positive samples vector
-h	height of positive samples vector
-bt	boosting type, Gentle Adaboost, LogiBoost, Real Adaboost, Discrete Adaboost
-minHitRate	true detection rate required in order for training to continue to next stage
-maxFalseAlarmRate	maximum false alarm rate per stage
-maxDepth	maximum tree depth
-weightTrimRate	pruning threshold, any features below this threshold will be disregaurded in each step of training
-mode	if images are symmetric, will only train on one half of image to reduce training time.

During the training process some data is displayed showing the progress of the training process. An example of this is shown in the figure below.



```
===== TRAINING 20-stage =====
<BEGIN
POS count : consumed   4559 : 5021
NEG count : acceptanceRatio    9080 : 9.96575e-007
Precalculation time: 19.884
+----+---------+---------+
|  N |    HR   |    FA   |
+----+---------+---------+
|   1| 0.999342| 0.985903|
+----+---------+---------+
|   2| 0.995394| 0.895595|
+----+---------+---------+
|   3| 0.995174| 0.776211|
+----+---------+---------+
|   4| 0.995174| 0.734361|
+----+---------+---------+
|   5| 0.995174| 0.588877|
+----+---------+---------+
|   6| 0.995174| 0.535793|
+----+---------+---------+
|   7| 0.995174| 0.418282|
+----+---------+---------+
```

Figure 6-12 Haar training output – Stage 20

The output displays the Hitrate(HR) and the False Alarms (FA) at each iteration of the training algorithm. As mentioned before this is repeated until the desired HR, and FA rate are achieved. In this thesis a 20 stage classifier was used for each object class. This is the most common size of haar classifiers used in most applications. Training on a 3Ghz Intel Core i7 processor took about 2 days to train a classifier.

When the training is complete, a classifier cascade Extensible Markup Language (.xml) file is generated. This .xml file stores all of the cascade information and is used to load the classifier for object detection.

6.3 Testing and Results

The training process of the haar classifier has many parameters which can dramatically affect the performance of the classifier. In order to establish the ideal parameters a study was conducted to establish the optimum parameters for the haar classifier for use in building and intersection detection.

6.3.1 Feature type

The openCV library includes two feature types to use with the cascade classifier detector. The most common is the Haar features. However another less well known feature type can be used, this alternative feature type is known as the Local Binary Pattern feature ,or LBP. The LBP features have traditionally been used for texture description, however recently in 2006 LBP were used in a cascade classifier for face detection [28]. The system is the same as described previously in this chapter; however the features used are slightly different. Haar features simply look at rectangular regions and from the sum of pixel intensities matches an appropriate haar like feature. Local Binary Patterns also look at square regions of the image, and approximate the weighted pixel intensity for each region. An example of the two different features is shown below. The LBP features for a face are shown on the left, and the haar like features are shown on the right

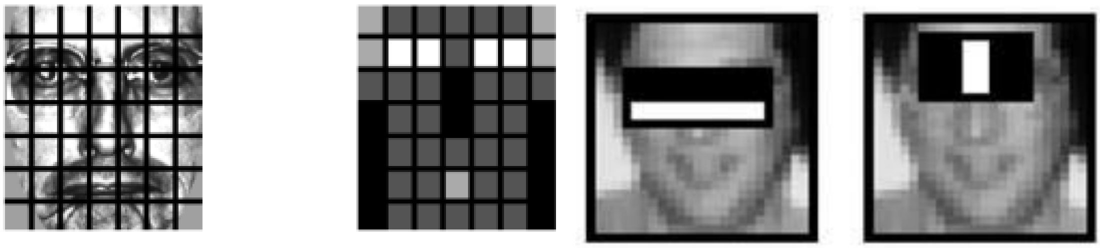


Figure 6-13 Local Binary Pattern features(left), Haar like features (right)

The main advantage of using LBP features is the much faster computation time. For this reason a test was conducted to find which feature type would be most suitable for aerial object detection.

Two classifiers were trained using Gentle AdaBoost methods with the same parameters, the only difference was the feature type used. The same training dataset was used for both the classifiers. The performance of the classifiers were tested on a variety of different aerial images.

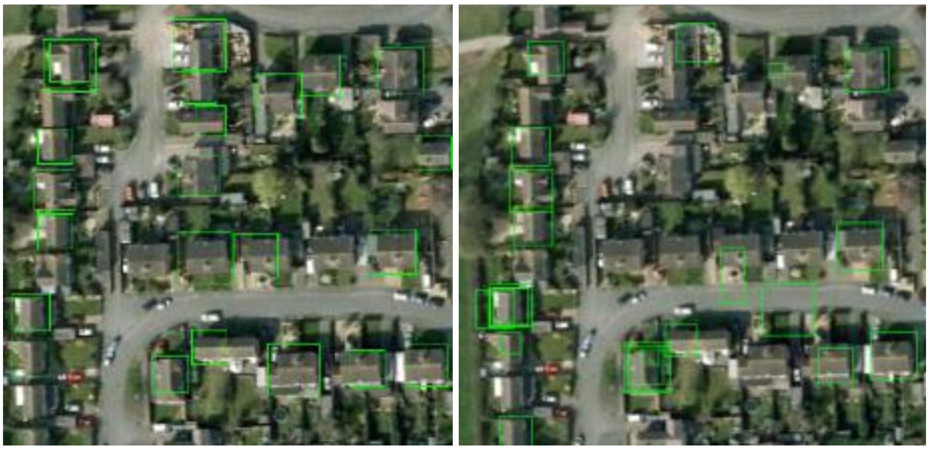


Figure 6-14 building detection using Haar features (left) and LBP features(right)

Figure 6-14 shows a comparison of the detection results for the two feature types. Although the LBP were computed about 50% faster than the Haar features, the result was much worse. Not only were fewer buildings detected, there were also a large amount of false detections.

These tests were performed over 10 different image regions, a total of 318 buildings were counted by hand in the images. The results are presented in terms of true and false detection rates below.

Table 6 Feature type comparison

Feature Type	True Detection %	False Detection %
Haar Like	81.85	12.91
Local Binary Pattern	53.09	32.77

Although the LBP feature set is run much faster than the Haar feature set, the accuracy is not very good. For this reason the Haar like features will be used as they are much more suited for detecting objects in aerial images.

6.3.2 Boosting Type

As mentioned previously, there are many different types of boosting algorithms that can be used to train the cascade classifier. The main boosting types compared are Gentle Adaboost, Real Adaboost, and Logit Adaboost. All the different training methods produce a cascade classifier; however the methods differ in some ways. The full details of each of the training methods can be found in [29] and [30].

The main purpose of this test is to establish which of the booting types are most suited to detecting objects in aerial images. Therefore four classifiers were trained, each one with a different boosting type. Each of the classifiers were trained using the same training dataset, and tested on the same images. A receiver operating curve (ROC) was plotted to help evaluate the performance of each training method. This method is often used to show the performance of object detectors. A line of y=x is the chance line, anything plotted above this line means that the detection method is better than random chance, the more the data fits to top left, the better the accuracy of the detection method. The ROC curve for the 3 different boosting types is presented in

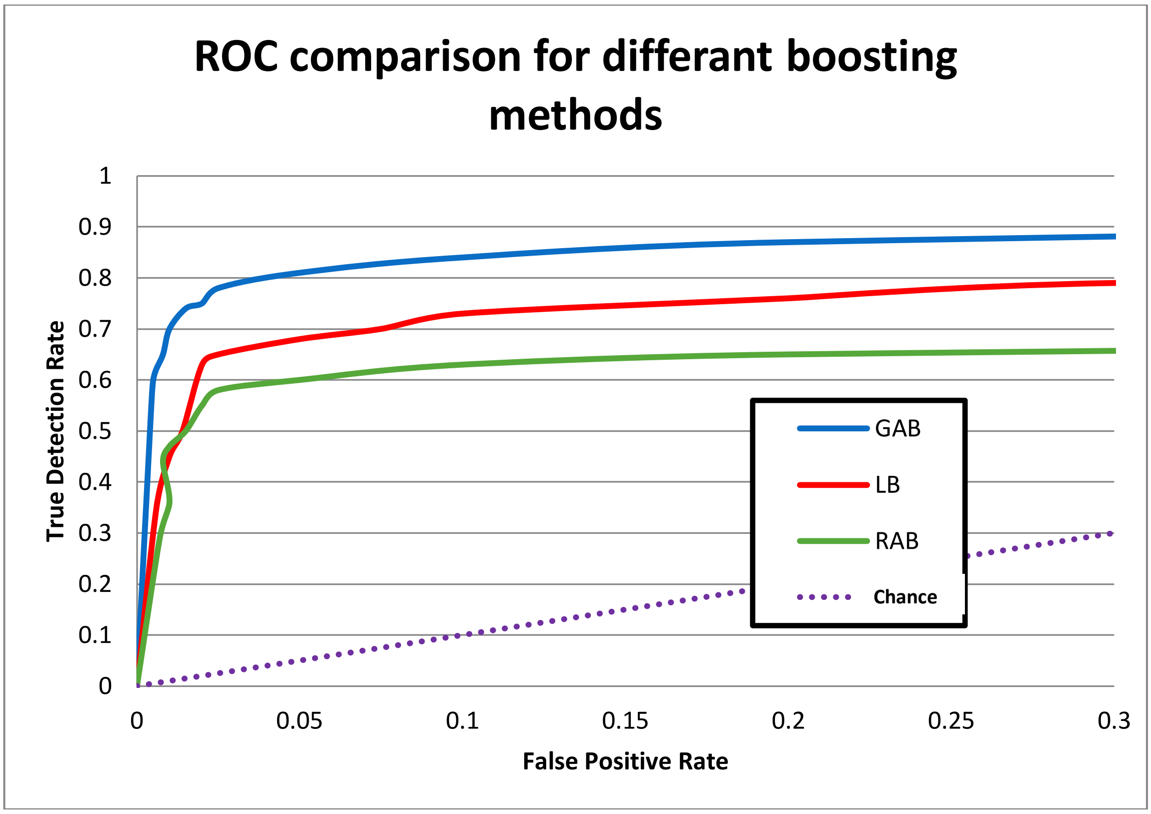


Figure 6-15 ROC curve for different boosting types

As Figure 6-3 shows, the Gentle Adaboost (GAB) training method outperforms the other methods. Although all the different classifiers used to the same number of training data, and training samples the results are very different. All three methods produced acceptable results (all fall above the chance line). The Gentle Adaboost method puts less weight on outlier data and this makes it more suitable to regression data. LogitBoost is similar to Gentle Adaboost and therefore has similar performance. The Real Adaboost method produces the worst results, this training method is based upon confidence rated predictions and is best suited to categorical data. Although there are a few main categories of buildings/ intersections, they are all relatively similar in terms of haar like features and for this reason the RAB method does not perform as well as the others. The RAB method may be more suited with datasets that have stronger categorical data.

6.3.3 Robustness Test

It is important to test the Haar classifier performance in the presence of varying image conditions. The classifier should be able to detect buildings/road intersections in a variety of different conditions that could be caused by changes in illumination, or seasonal effects. Several different test regions were chosen, for each region, images were obtained from different dates. Some of the examples are shown below.

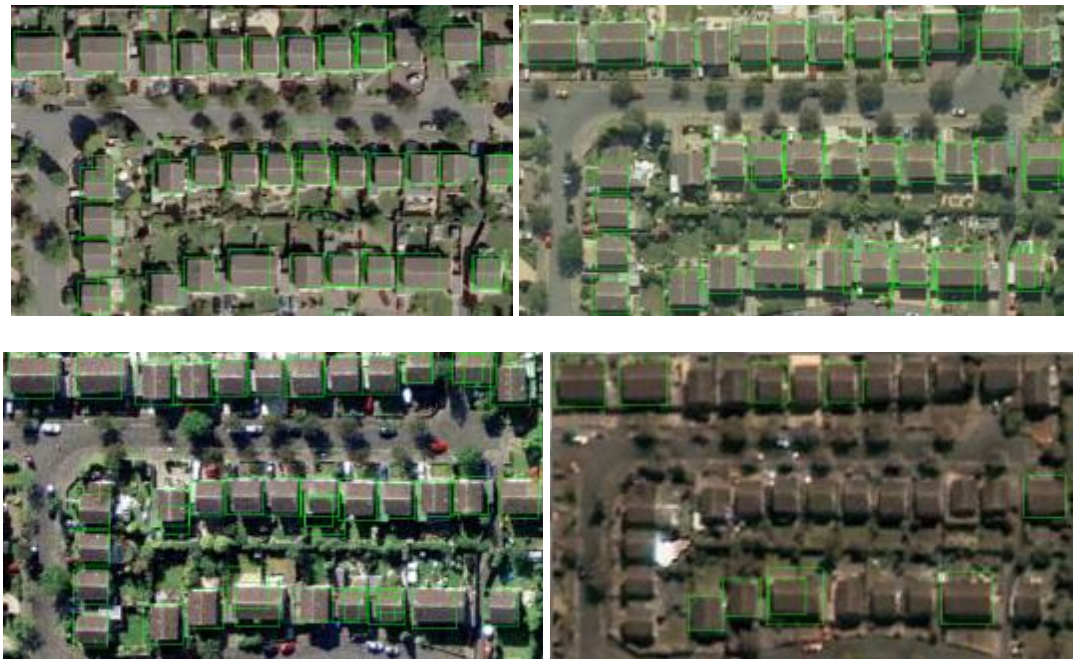


Figure 6-16 Building detection under different conditions

As shown above in Figure 6-16 the Haar classifier is able to robustly detect buildings under different conditions. An interesting case is that of the image in the bottom left. This image was the oldest available and was taken under very poor conditions, which resulted in a underexposed, and grainy image. Despite the poor quality of this image the Haar classifier was still able to detect many of the buildings. When the images were of acceptable quality the classifier did not have any difficulty correctly detecting the buildings.

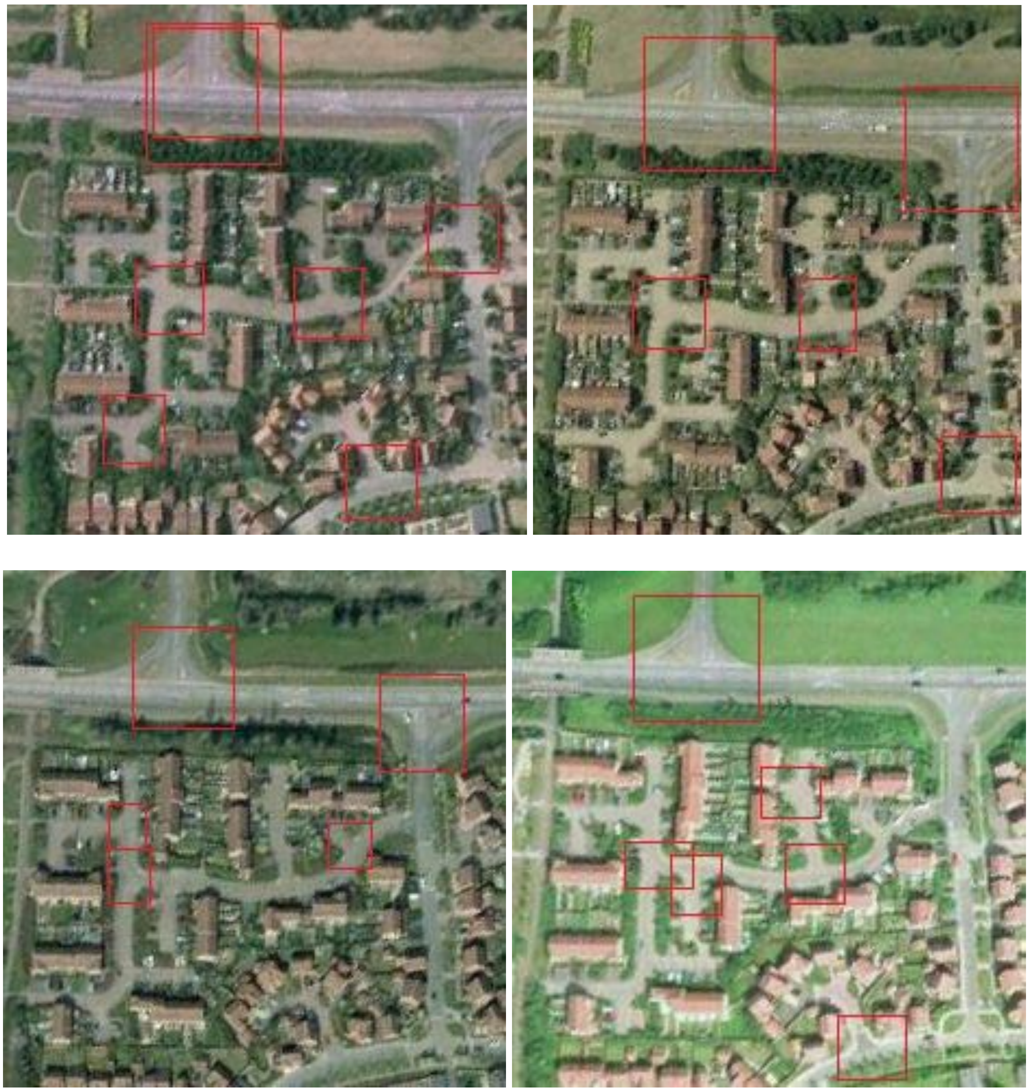


Figure 6-17 Intersection detection under different conditions

Figure 6-17 shows some of the results of the intersection detection under different image conditions. Most of the same intersections are detected between all of the different conditions.

The tests conducted have shown that the haar classifier is able to deal with different image conditions; this is largely due to the fact that buildings and roads do not vary very much. More testing was not possible due to no sample data on images under different seasonal conditions such as snow. But in general the haar classifier is able to deal with different image conditions as shown and this means that the reference database will not need to updated on a regular basis.

7 Comparison of Detection Methods

This chapter will compare the performance of each of the different landmark detection methods across a standard dataset. This will allow for a thorough comparison of how each detection method compares to one another.

In previous chapters, two main methods for building detection were implemented (Edge based Building, Haar classifier), and two main road intersection detection methods were implemented (line based road detection, and Haar classifier). Although the haar classifier is capable of detecting both roads and buildings it is still important to directly compare all the the different methods.

This chapter tests each detection method on the same dataset of images, and compares the two road, and building detection methods.

Test conditions

A collection of aerial images were chosen to investigate how the detectors perform under specific situations. A range of images where chosen or intentionally degraded to simulate some of the conditions the landmark detectors will need to deal with.

- Blurring
- Perspective
- Rural
- Suburban
- Urban/ City center
- Warehouse/ unique buildings
- Poor Exposure
- Low resolution
- Different Altitude

The results and discussion for the building detection methods are shown in section 7.1 and the road intersection detection results are presented in section 7.2. The results are presented in terms of true detection rate and false detection rate.

7.1 Building Detection Comparison

This section looks at performance of the two different building detection methods. Figure below shows the overall performance under the various conditions for each of the building detection methods. Further discussion and example images for each of the test condition are shown later in this section.

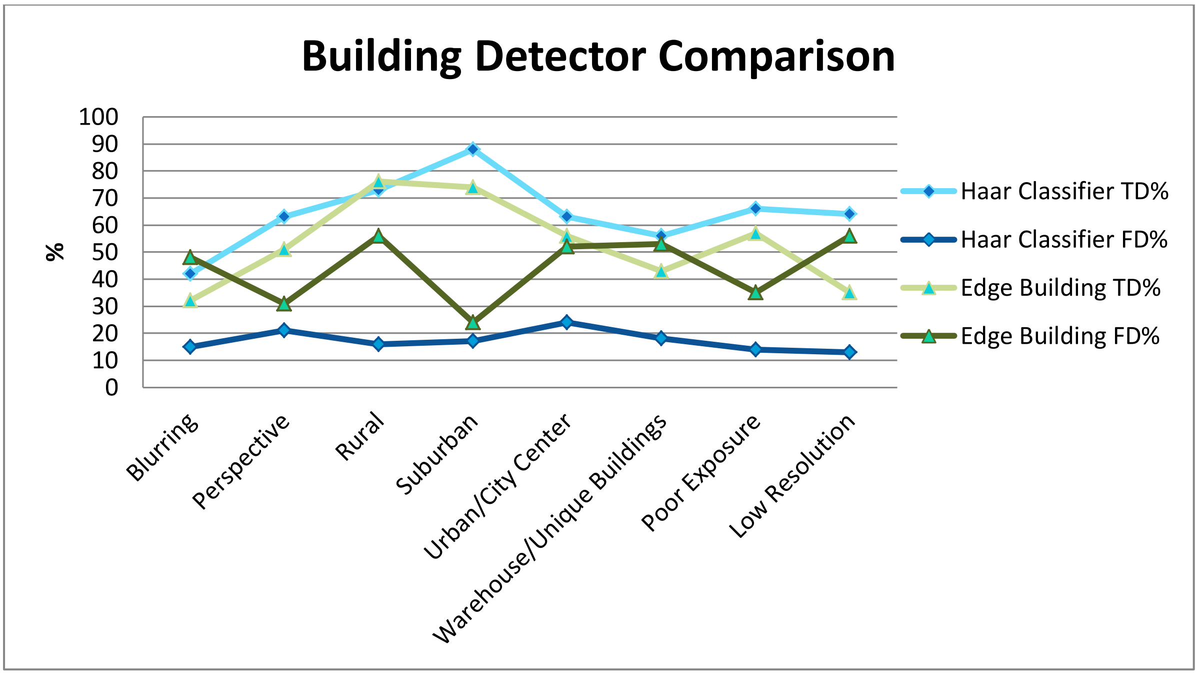


Figure 7-1 – Graph showing the building detection method performance in terms of True Detection (TD) and False Detection (FD) rates

It can be seen that in general the Haar Classifier outperforms the Edge based building detector in terms of both true detection (TD) rate and low false detection (FD) rates. The edge based building detector always has a much higher false detection rate. This high false detection rate will make things very difficult when trying to match the landmarks onto a database as the majority of landmarks will not correspond to a position on the map so the system will have a very low certainty of a correct match. Furthermore it can also be seen that the best results were found for suburban regions for both detectors. This is mainly because this was the case both detectors where designed around. Suburban areas also usually have buildings that are detached which make them much easier to detect and distinguish from other regions.

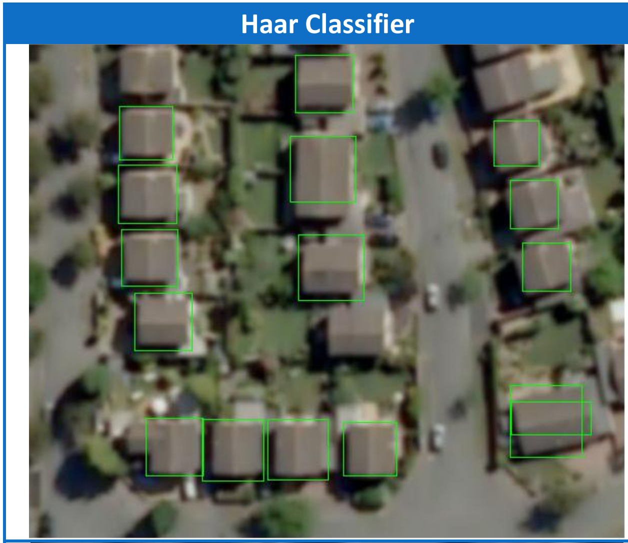
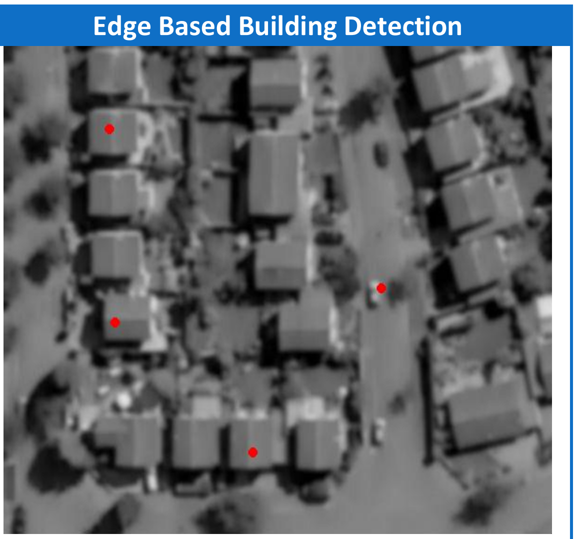
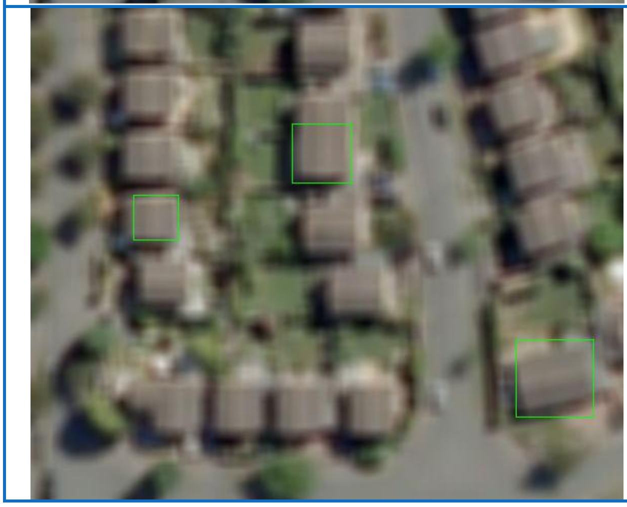
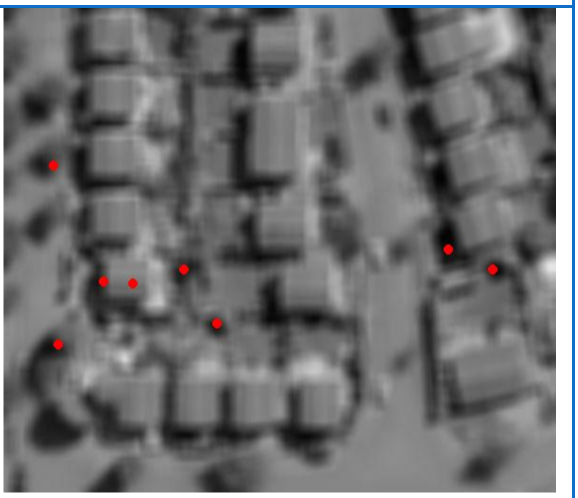
The robustness of the haar classifier is clear as it is capable of detecting the buildings under a variety of difficult situations such as high blurring, poor exposure, and low resolution, all while keeping the number of false detection under 20%. The edge based method did not perform very well under such conditions, and sometimes the FD was higher than the TD rate which would makes it unsuitable to be used for such situations. The main reason for the poor performance under the blurring tests, is that when an image is blurred, the edge information is broken down as the image is essentialy smoothed.

Example images and some more discussion for each of the test conditions are shown below. For all the images the Haar classifier results are on the left, and the edge based building detection results are presented on the right. Approximately 5 images where tested for each condition, however only a few of the more interesting results are shown and discussed below.

Blurring

During bad weather or during aggressive maneuvers, the camera gimbal may not react fast enough to cope with the aircraft movement. Other factors such as an out of focus camera, or dirty lens may cause some blurring in the image. It is useful to see how the two detection algorithms handle the blurring effect. An aerial image was manually blurred, using simple Gaussian blurring of increasing intensity. The results can be seen below.

Table 7 - Building detection comparison examples for blurry conditions

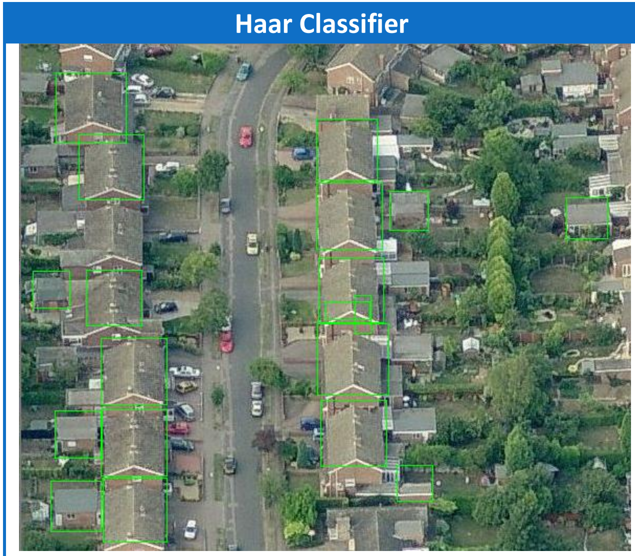
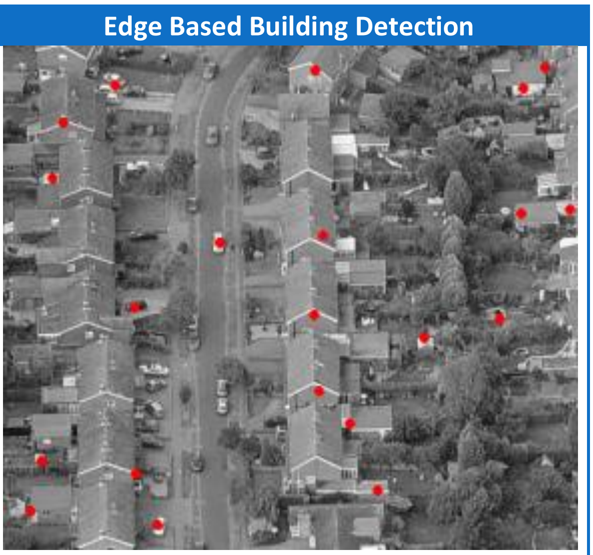
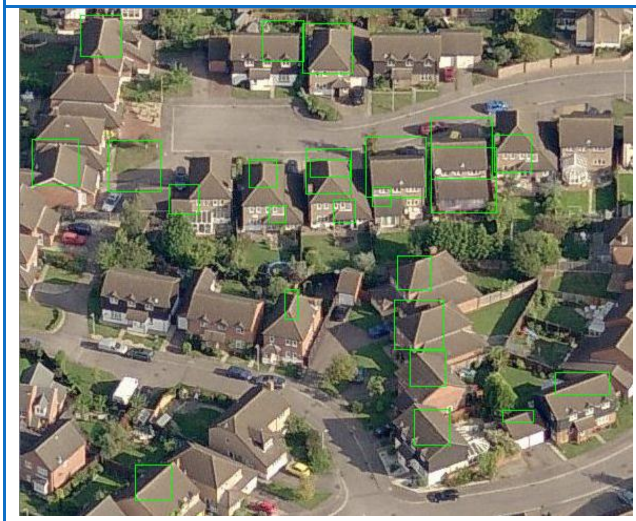
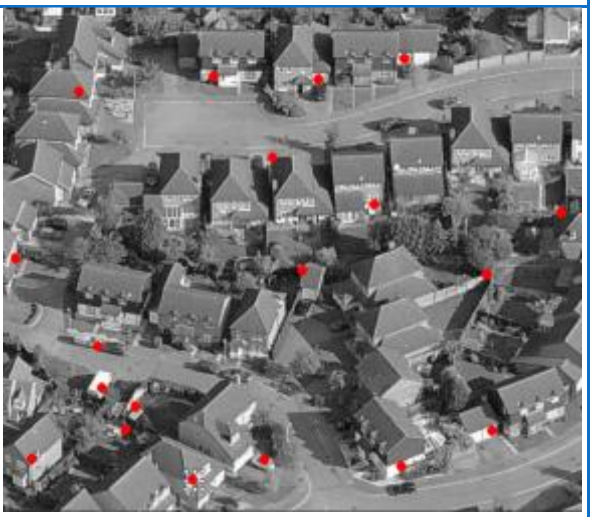
Haar Classifier	Edge Based Building Detection
	
	

The haar classifier can handle moderate levels of blurring, and in general has much better performance compared to the edge based building detector. The main reason for the poor detection rates of the edge based detection method is that when an image is blurred the edges in the image are smoothed. Because the edges are weaker, the bilinear filter process further reduces the image and as a result the building edges are not very prominent and difficult to find.

Perspective

Again if the aircraft is flying in rough weather, or performing aggressive maneuvers, the camera gimbal may not react fast enough or a camera gimbal may be broken/ not used. It is useful to see how the detectors handle situations when the buildings are not viewed exactly from above. Aerial images taken at a 45^{o} angle where used.

Table 8 - Building detection comparison examples for images that are taken at 45^{o}

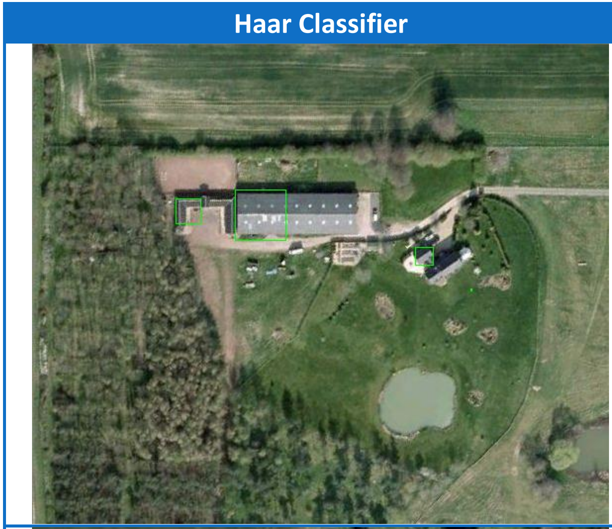
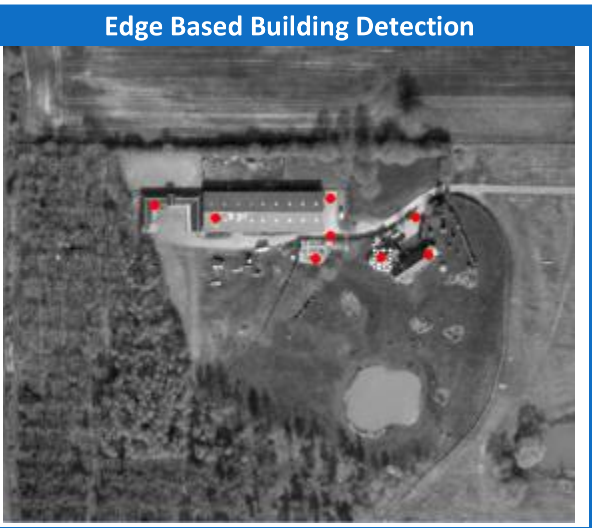
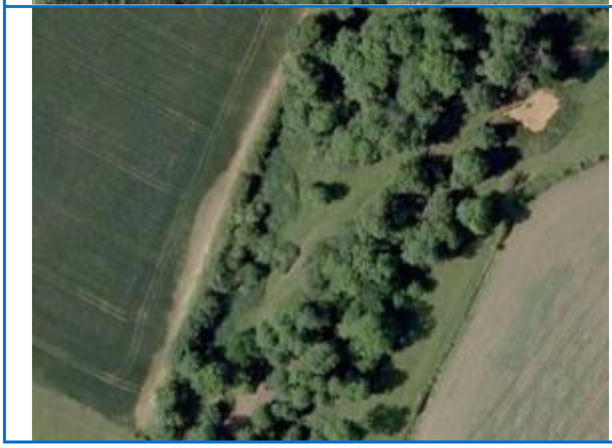
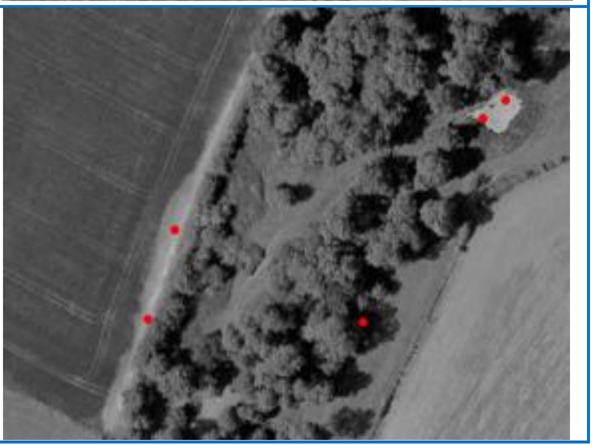
Haar Classifier	Edge Based Building Detection
	
	

Overall both detectors still perform quite well, this is because the roofs of the buildings can still be seen and are therefore correctly detected. The edge based building detector found it difficult to accurately detect the buildings as when the sides can be seen, they usually contain stronger edge information and therefore most of the detection points are on the sides of the buildings, opposed to the actual roof.

Additional processing will be required to geo-locate the positions of the buildings when the camera is not looking directly down, but this comparison shows that the haar classifier can still detect buildings fairly well that are not viewed exactly from above.

Rural

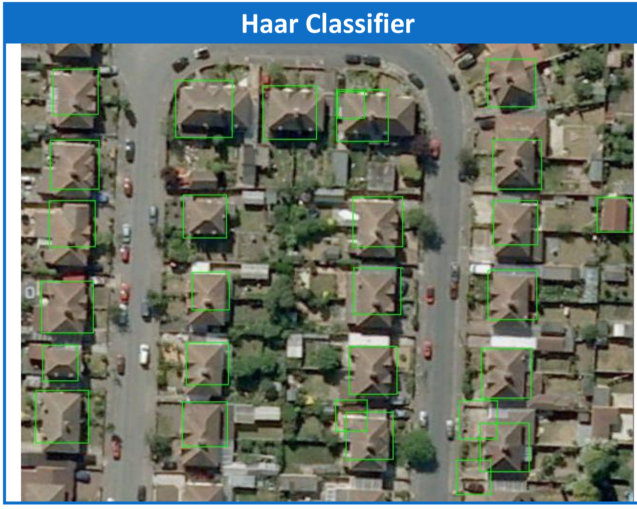
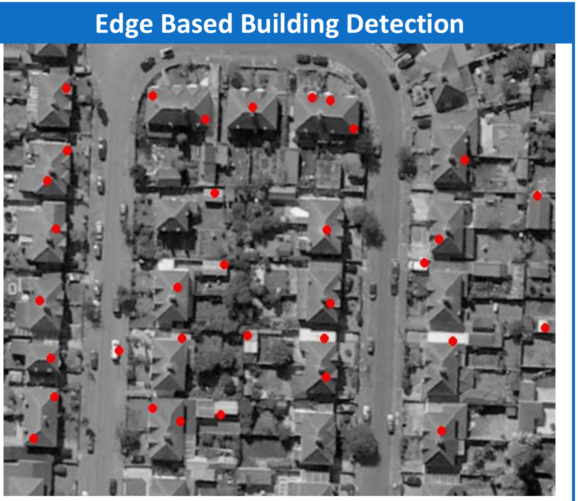
Table 9 - Building detection comparison examples for rural regions

Haar Classifier	Edge Based Building Detection
	
	

The edge based building detector, although producing a higher true detection rate, also suffers from a higher false detection rate, this is particularly true when flying over regions that do not contain any buildings. Due to the probability density function assuming that there is always at least once building in the image, it will take the strongest response and take that to be a building. All of the other responses will be compared to the strongest detection. When flying over regions with no buildings, the detector will take regions that would not have a strong edge response and assume them to be a building. This causes a high number of false detections when there are no buildings present.

Suburban

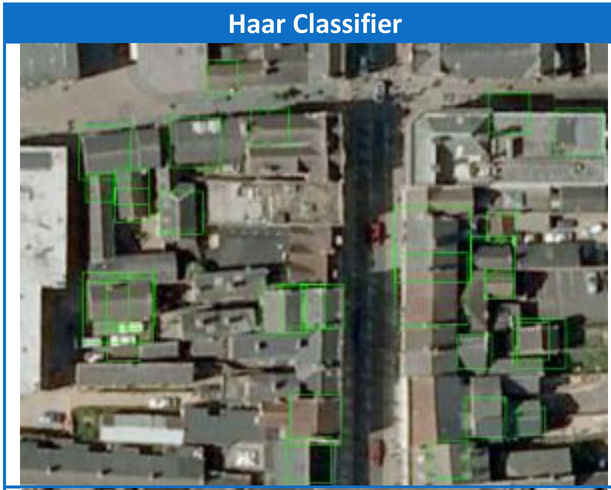
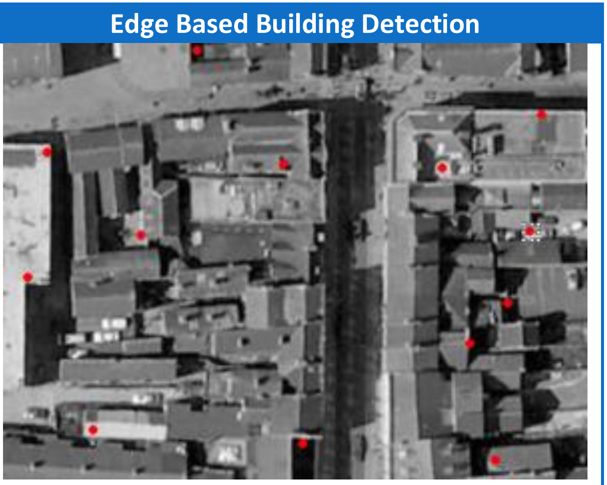
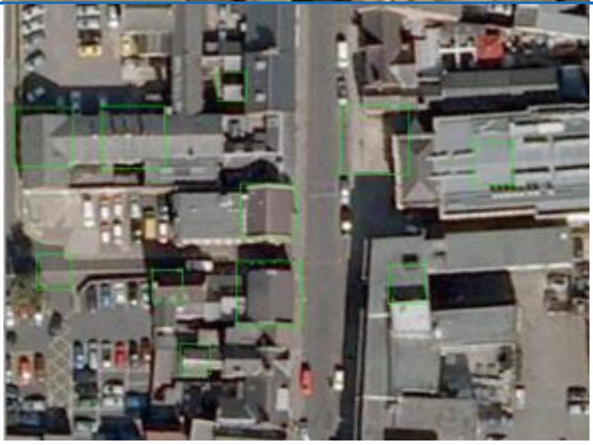
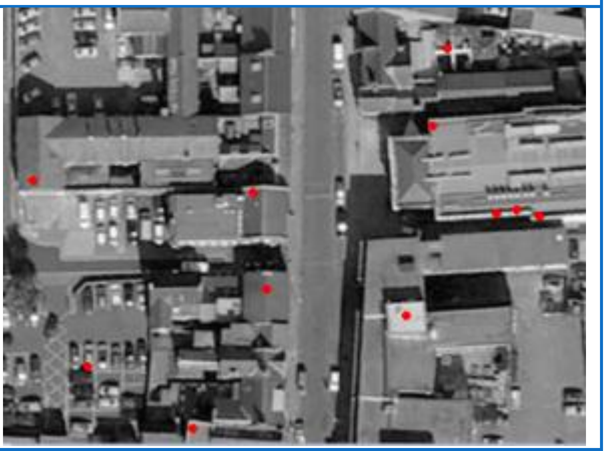
Table 10 - Building detection comparison examples for suburban regions

Haar Classifier	Edge Based Building Detection
	

This is the region that the detectors where designed for and as a result have the highest performance in suburban environments.

Urban/City Center

Table 11 - Building detection comparison examples for urban regions

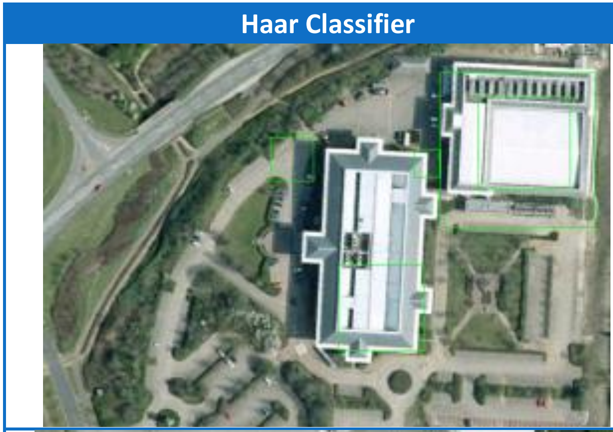
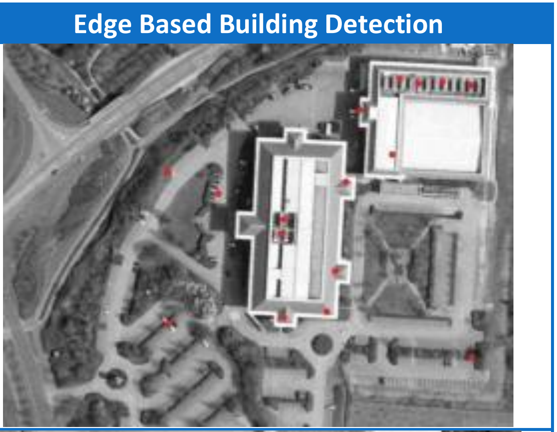
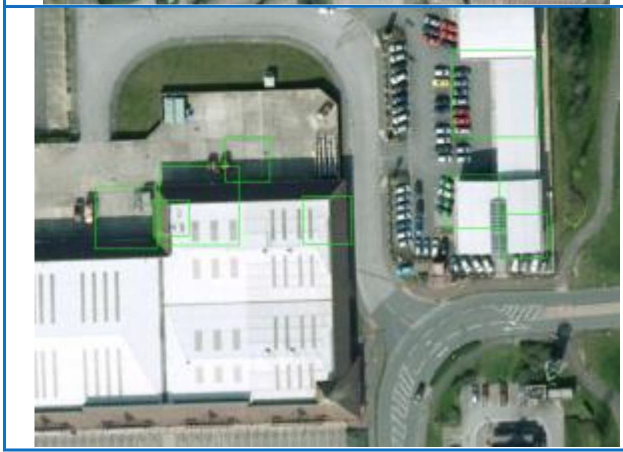
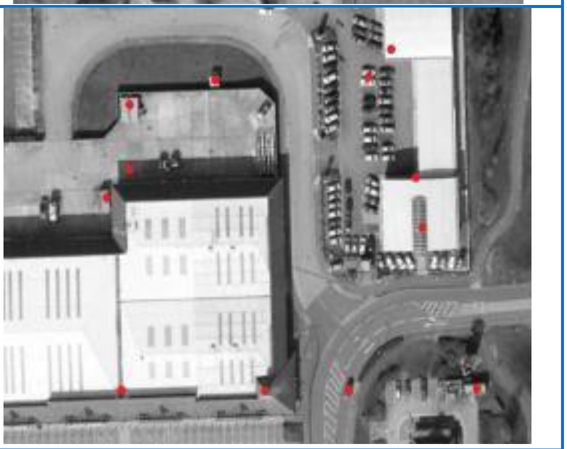
Haar Classifier	Edge Based Building Detection
	
	

It can be seen that in the images above, it is still difficult for a human to detect all of the buildings correctly. However the haar classifier is still able to detect a high number of the buildings with a fairly low false detection rate. Overall the haar classifier outperforms the edge based method. The performance of the haar classifier could be improved further by including additional training data for urban region buildings.

Warehouses/ unique buildings

Images that contained unique buildings like warehouses or office blocks were compared using the two building detection methods.

Table 12 - Building detection comparison examples for unique buildings and warehouses

Haar Classifier	Edge Based Building Detection
	
	

It can be seen that both methods do not detect the unique buildings very well. Only parts or corners of the buildings are detected.

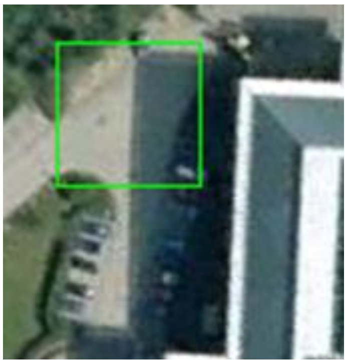


Figure 7-2 Shadow from building detected as a building roof.

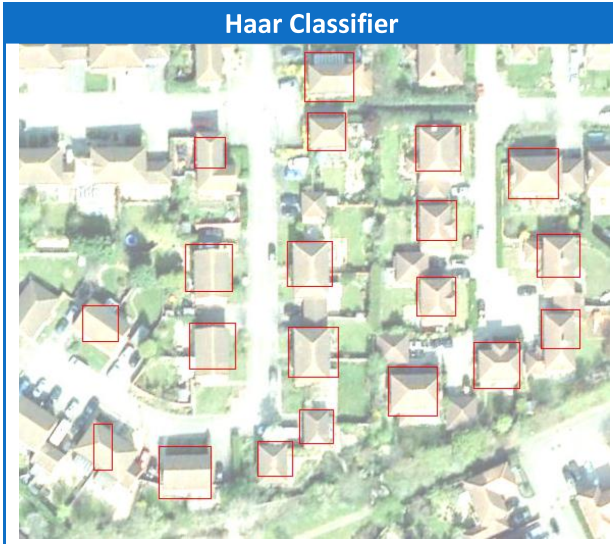
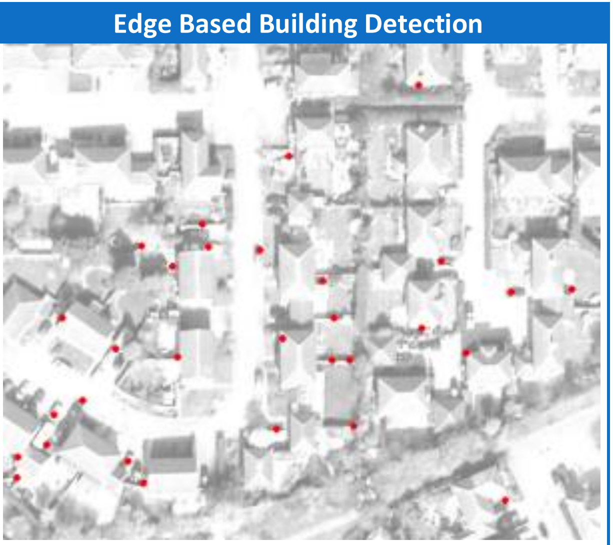
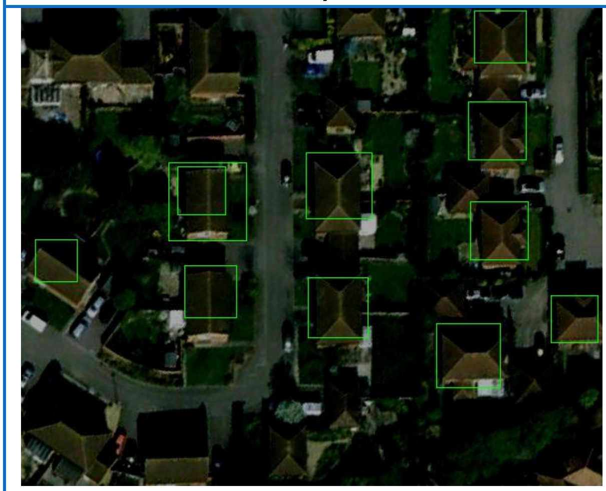
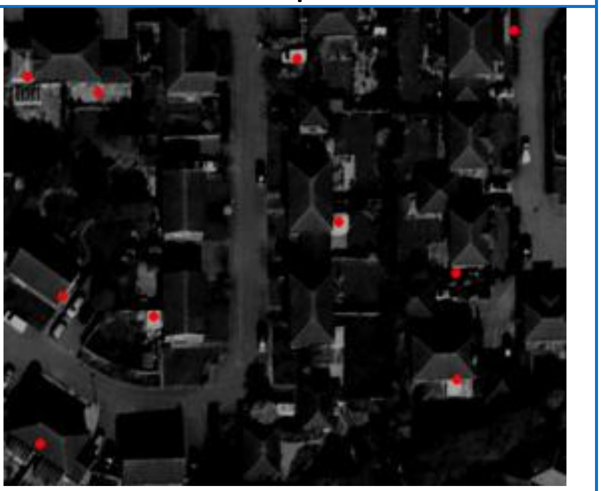
As shown to the right, sometimes the haar classifier detects a shadow region as a building roof. It can be seen that this shadow region does infarct look like an actual building roof so it is very difficult to deal with such a situation. However these cases do not occur very often, and when they do they usual fall on a smooth regions next to tall buildings.

The nature of the haar classifiers means that they require training, and as described in previous chapters, the building detection was trained using typical residential buildings, therefore when the classifier encounters unique buildings that have not been included in the training dataset they do not perform well. However despite this the general shape such as building corners are still detected by both methods. Also the large smooth regions of the warehouse buildings mean that the edge based building detector only can detect the corners of the large buildings.

Poor Exposure

On occasion the camera onboard may take a poor image due to over or under exposure, this could happen when clouds suddenly cover the region and the camera could take some time to adjust the exposure and white balance. This effect is not likely to occur if a high quality camera is used, as this effect is more common on the cheaper, low quality camera models.

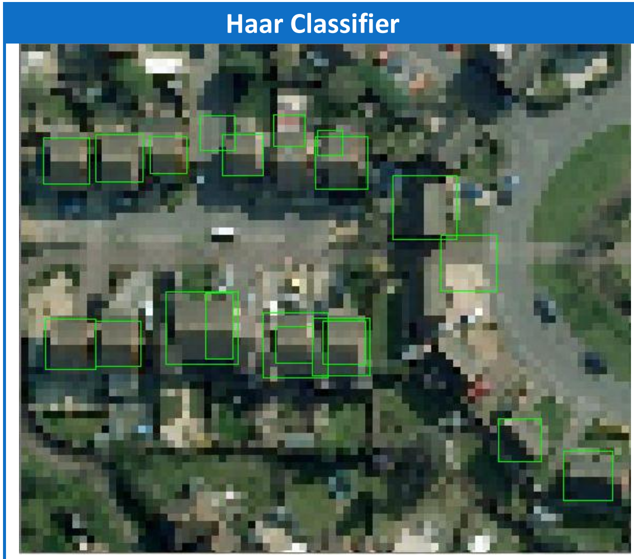
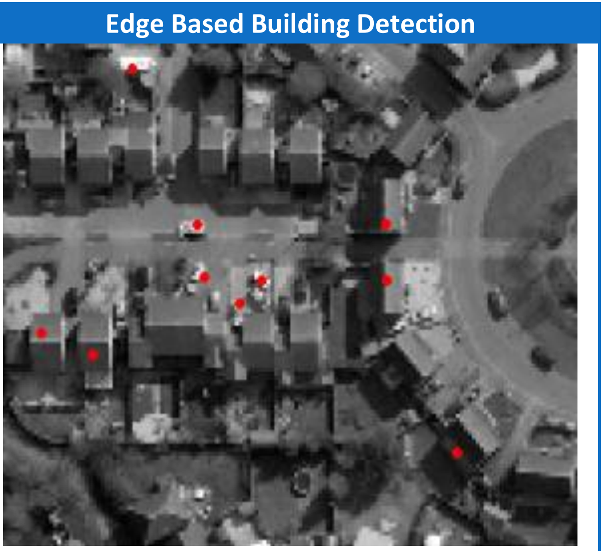
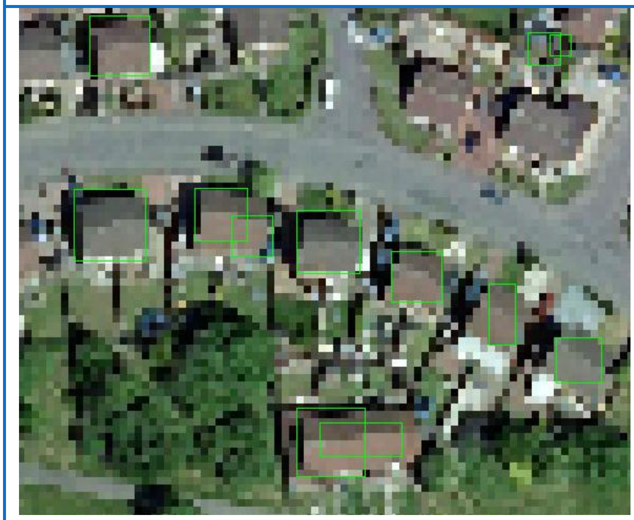
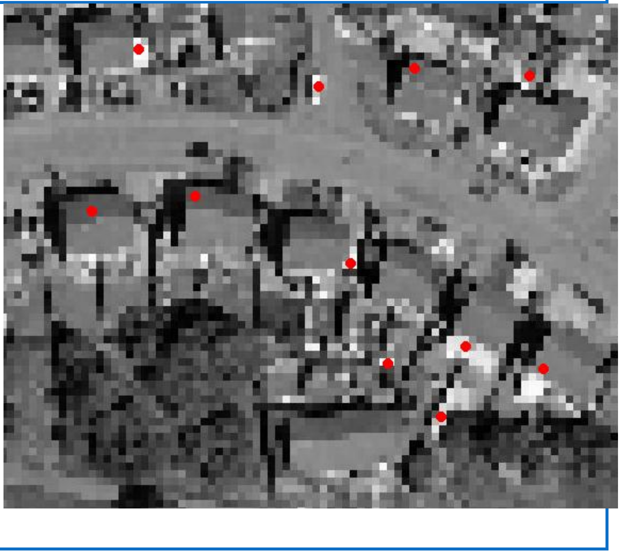
Table 13 - Building detection comparison examples for poor exposure images

Haar Classifier	Edge Based Building Detection
 **Over Exposed**	 **Over Exposed**
 **Under Exposed**	 **Under Exposed**

Both the detection methods can detect buildings better when an image is over exposed. When the image is under exposed and very dark the detection performance is not as good because the dark building roofs do not stand out very well. The edge based detector does not detect dark roods very well, but they can detect lighter regions quite well despite the under exposed conditions.

Low Resolution

Table 14 – Building detection comparison examples for low resolution images

Haar Classifier	Edge Based Building Detection
	
	

Some low resolution images where tested to see how the detectors would perform at different altitudes. At higher altitudes the spatial resolution would be lower so each building would be fewer pixels. Once again the haar classifier detects the buildings more accurately with less false detection.

7.2 Road Detection Comparison

This section looks at the two road intersection detection methods. Only some of the test images are displayed below to show some of the examples of the detector performance under different conditions.

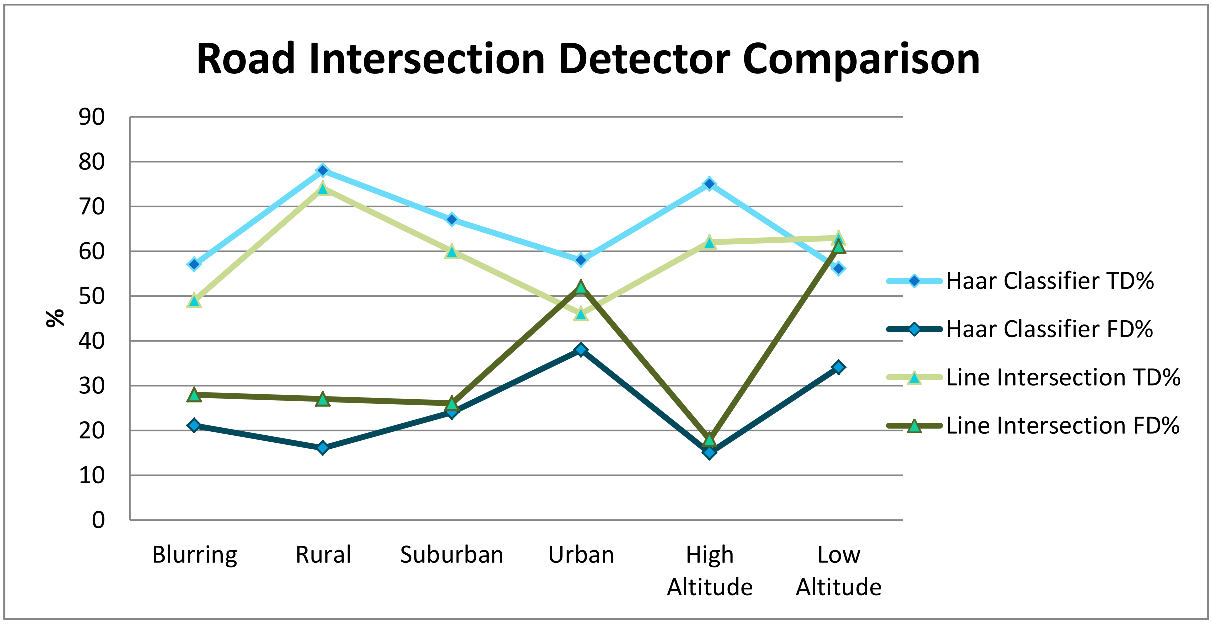


Figure 7-3 – Graph showing the Road intersection Detector comparison across the different environments in terms of true detection (TD%) and false detection (FD%)

As shown in the graph above, the haar classifier generally outperforms the other line based intersection detector. Both methods have fairly high detection rates in rural conditions, this is mainly due to the fact that roads are fairly distinct and there is less noise in rural images. The line based method still has a higher number of false detection in rural regions and this is mainly due to other linear features such as hedges that sometimes cause the detector to falsely detect road intersections.

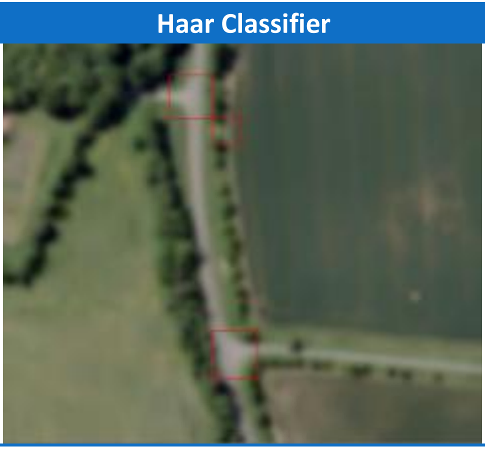
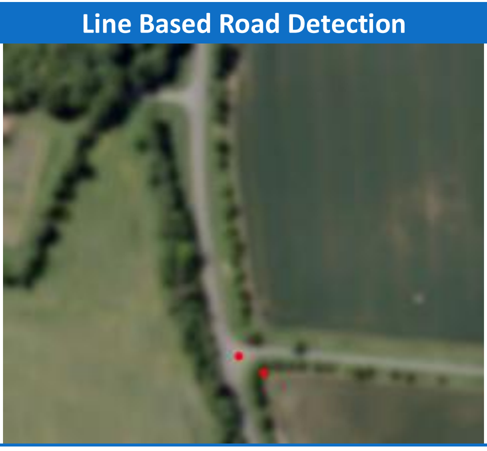
Blurring has a negative effect on both detection methods, this is mainly due to the fact that the linear regions are less clearly defined due to the smoothing so the accuracy of the curvilinear detector and the haar like features are reduced since these regions are harder to find.

Urban regions proved to be the most difficult regions to accurately detect road intersections. The reason for haar classifier is that most of the training dataset was obtained from suburban and rural road intersections; therefore the performance could be improved by adding more urban region road intersections as they are slightly different. The line based intersection detector also does not do very well under urban situations because there are many other long linear regions that have similar colors to roads, these are large office buildings that run parallel to the roads. These buildings are incorrectly detected as roads, and this causes many false intersections to be detected.

It can also be seen from the results that both detectors do not perform very well at low altitudes, this is because there is an abundance of information/noise in the image which makes it difficult to reliably extract the road regions. At higher altitudes, the road regions appear more smooth and distinctive and as a result both detectors perform much better at higher altitudes.

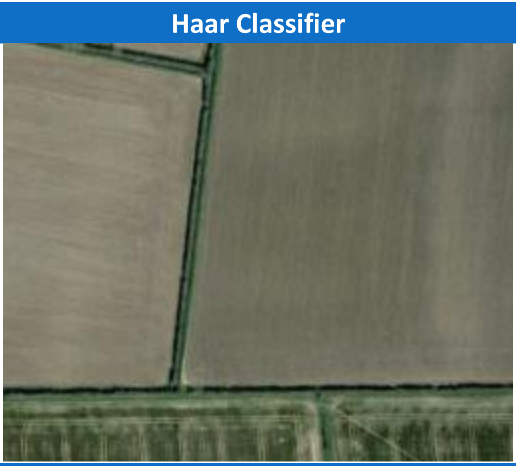
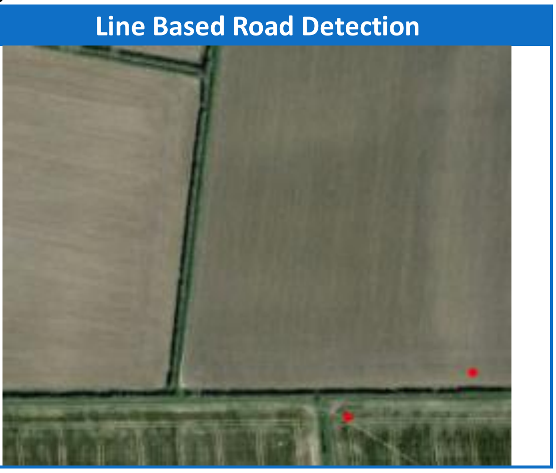
Blurring

Table 15 – Intersection detection comparison examples for blurry images

Haar Classifier	Line Based Road Detection
	

Rural

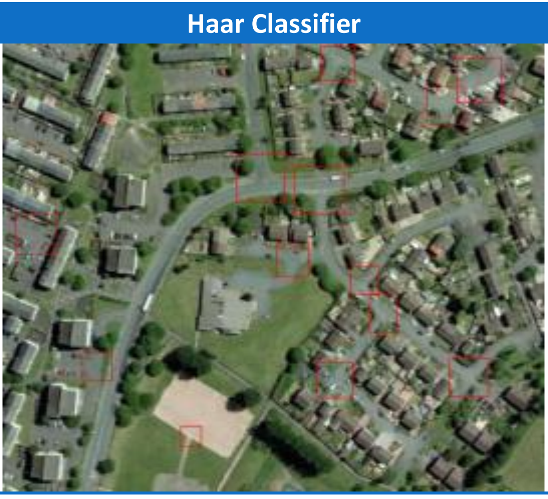
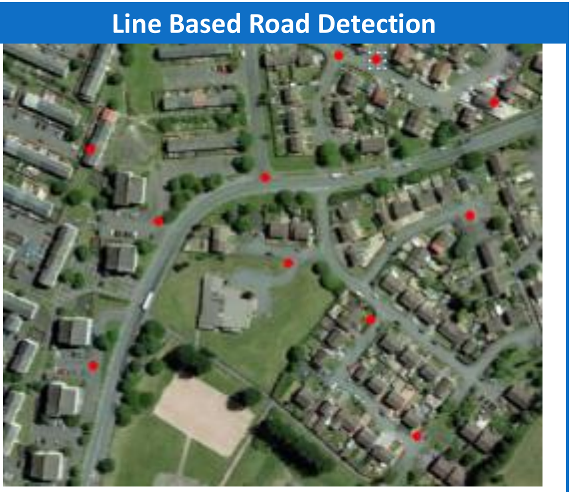
Table 16 - Intersection detection comparison examples for rural regions

Haar Classifier	Line Based Road Detection
	

The above image shows linear regions that could be misinterpreted as a road, particularly for the line based road detection method. The haar classifier does not have any false detections in such regions. However the line based detector has some trouble due to the window width and cannot correctly distinguish between some linear regions in the image and as a result, there are some false detections.

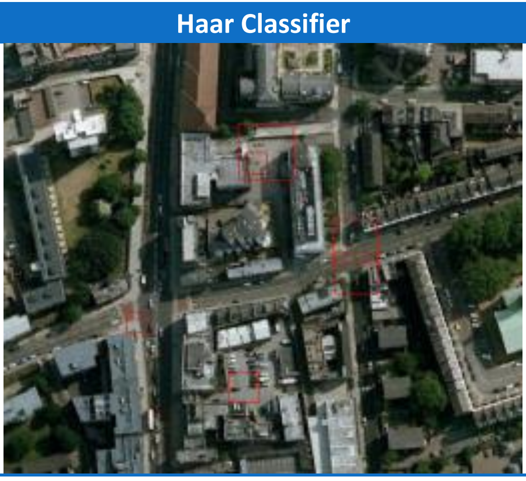
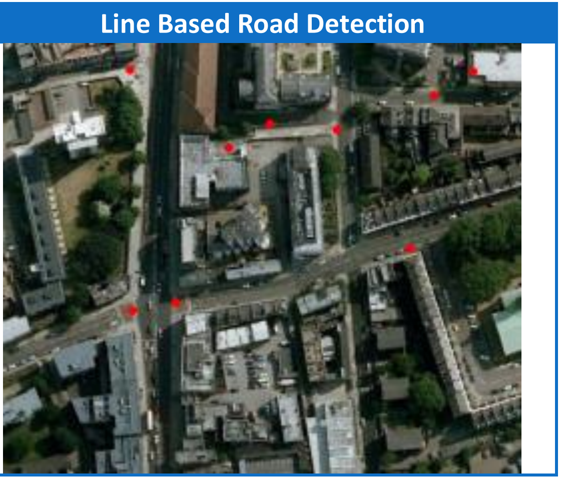
Suburban

Table 17 - Intersection detection comparison examples for suburban regions

Haar Classifier	Line Based Road Detection
	

Urban

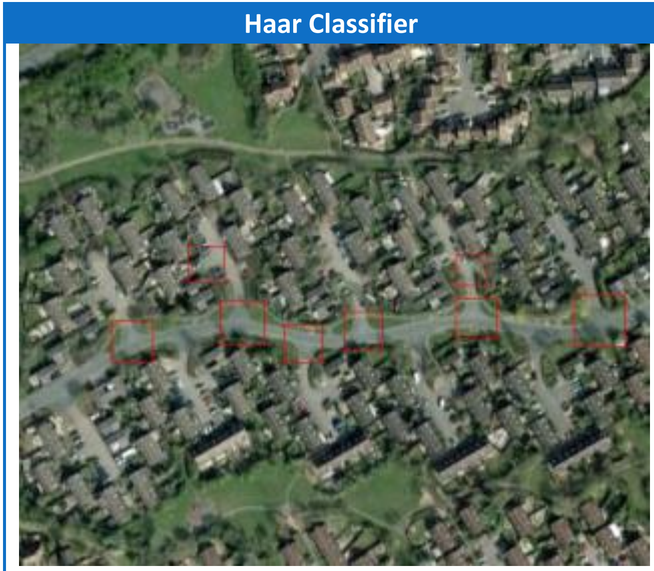
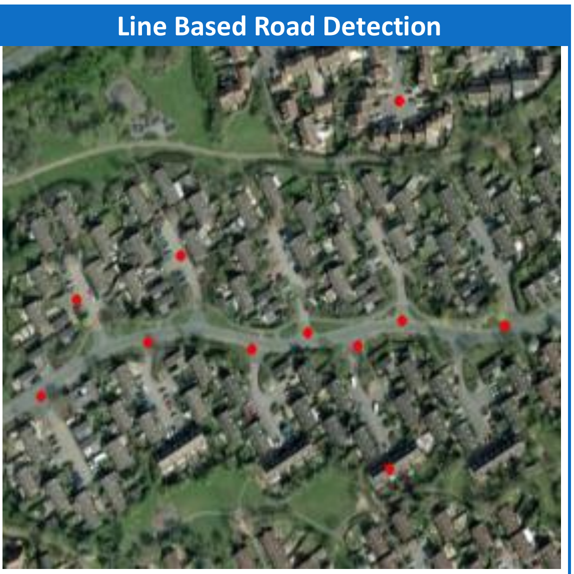
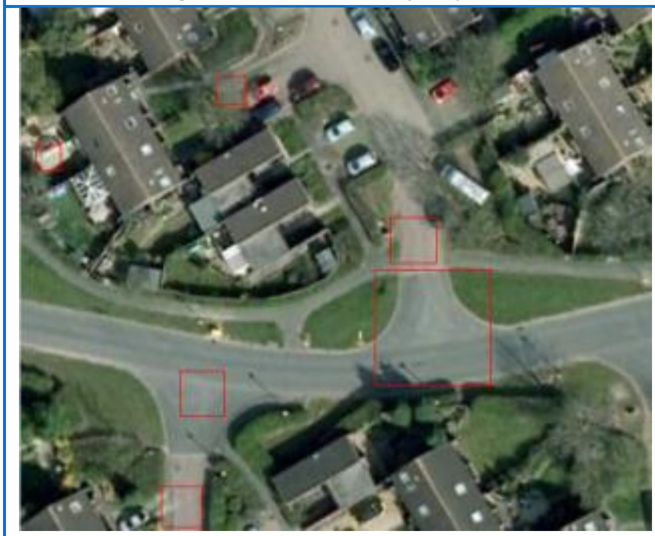
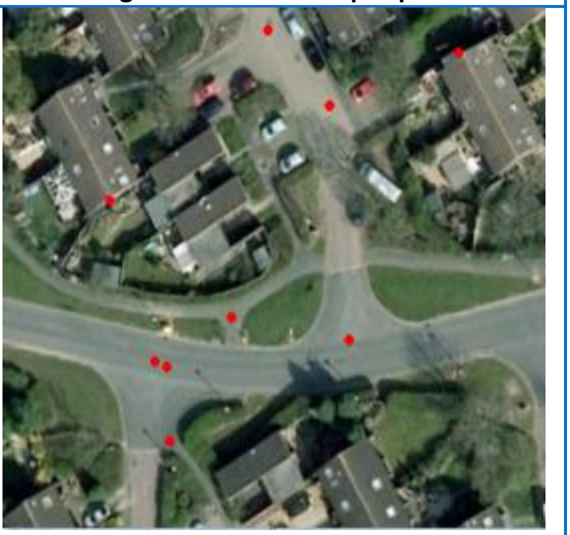
Table 18 - Intersection detection comparison examples for urban regions

Haar Classifier	Line Based Road Detection
	

In urban regions there are many large buildings that are sometimes connected, these buildings are often parallel to the roads and for this reason the line based road detection method finds it difficult to detect the intersections accurately. In addition the tall buildings in urban areas also create strong shadows; this causes the roads to appear less linear due to shadows that break up the road into smaller sections.   These shadows also make it more difficult for the haar classifier to detect road intersections.

Altitude

Table 19 - Intersection detection comparison examples for different altitudes

Haar Classifier	Line Based Road Detection
 **High Altitude – 1.3m per pixel**	 **High Altitude – 1.3m per pixel**
 **Low Altitude – 24cm per pixel**	 **Low Altitude – 24cm per pixel**

It can be seen that both detection methods perform better at higher altitudes. As mentioned previously, the main reason for this is that at lower altitudes there is much more detail in the image, this causes additional regions such as footpaths to be detected as roads. At low altitudes details such a lanes can also be seen on the road surface which can cause additional false detections. At higher altitudes the road regions appear more smooth with less detail, and for that reason they are detected much more accurately at higher altitudes.

8 Flight Case Study

Initial controlled tests have been performed using different methods and algorithms, as discussed in previous chapters. This gave a good approximation of which methods are most suited to the application of vision based positioning. The final step is to conduct a case study to assess the performance of the detector in a typical environment that the system will need to deal with.

Based on previous chapters the Haar classifier was found to be the most promising method for object detection. The initial controlled tests showed that the haar detector is capable to deal with difference scenes and different illuminations which is an important factor for real world use. The case study will test the detector in a typical mission the detector will encounter, to see how it performs under more realistic tests under varying conditions and environments.

8.1 Simulated Flight

The case study performed was to create a simulated flight path around the Milton Keynes and Bedford area using Google earth data. A flight path was chosen to cover a variety of different terrain, from rural areas, small villages and built up suburbs. A flight plan was created by creating a KML file. As this vision based positioning system is designed for use onboard a UAV with a MTOW of about 150Kg and a wingspan of 5m, a suitable flight path was created that would mimic a typical flight of such a UAV. The estimated speed of the UAV was chosen to be around 100kts. It is not important to accurately model the behavior of the UAV in this simulation as the camera will be on a gimbal which will keep the camera perpendicular to the ground.

8.1.1 Flight Route

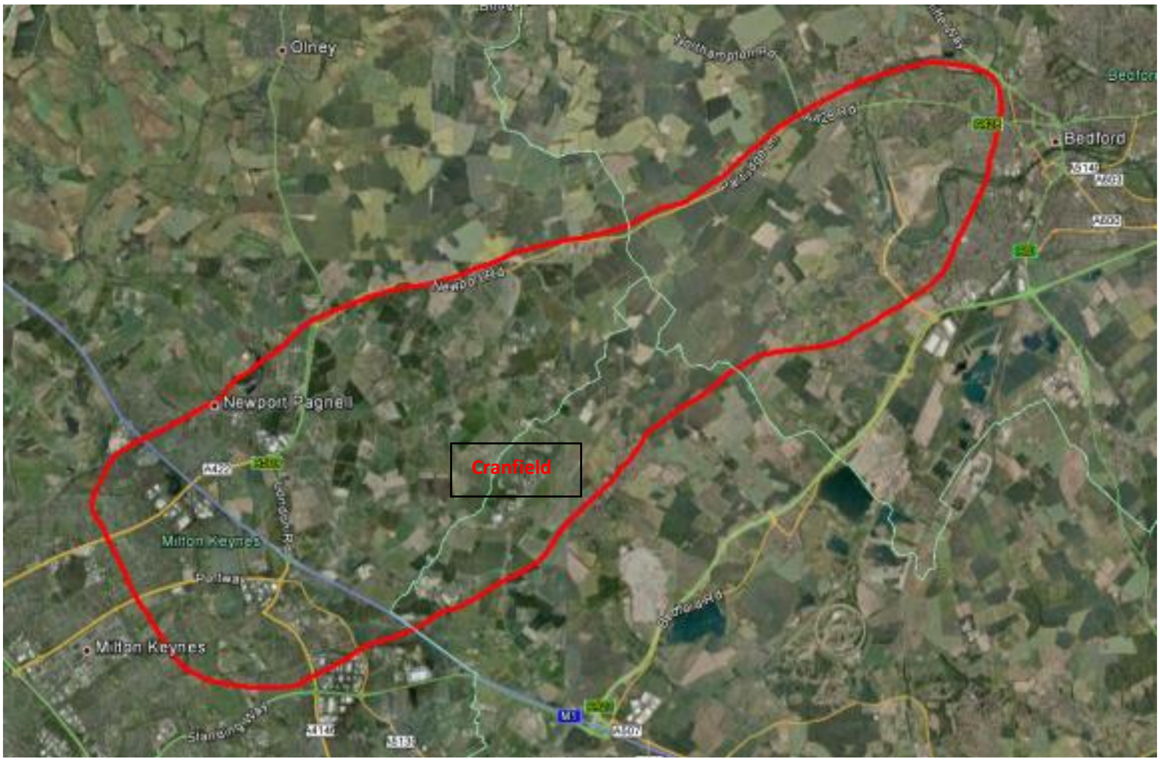


Figure 8-1 Flight Route used for testing

As shown in the image above, the flight route is a circular path that was chosen to fly over two major cities (Bedford, and Milton Keynes), but between the two cities the path follows some roads which pass through minor villages as well as small settlements and countryside. This flight path was chosen to go over a combination of both countryside and built up areas. Also some road regions are followed to enable the intersections to be detected. As the main purpose of this test is to test the Haar detector performance over a variety of environments, the flight speed was set to be approximately 100kts. Other flight dynamics such as turbulence, ignored. A screencast was captured of the flight in Google earth at 100kts and this video was then processed to detect videos. The altitude of the flight path was between 1500ft – 3000ft, which as mentioned before would be a typical flight altitude. The video stream was then processed to detect objects in each frame. In the real system the recorded video would simply be replaced by a live video feed onboard the UAV.

The building detector is computed at lower altitudes from 1500-2000ft, and road intersections are detected at the higher altitudes of 3000ft. The reason for only detecting road intersections at higher altitudes is that matching to a database cannot take place with only one intersection point. The matching system requires at least a few intersection points in order to establish a position estimate. Because road intersections occur less frequently than buildings, a higher altitude is used in order to capture more intersections on each frame.

8.1.2 Results and comments

Overall the Haar classifier is able to deal adequately with a variety of different environments that were encountered along the flight path. A summary of each of the detected objects are shown below in Table 20 across the flight path.

Table 20 Simulated Flight Detection Performance

Object	True Detection %	False Detection %
Building	81.30	14.91
Intersection	67.82	28.47

It can be seen that the building classifier has a slightly better true detection performance when compared to the intersection classifier. The main reason for this is because road intersections have more variation variety than building roofs, this problem could be reduced if more training samples were used to help deal with the large variety of road intersections. Another reason for the lower performance of the intersection classifier is that many of the road intersections in built up areas were partially covered by shadows from surrounding buildings as shown below.

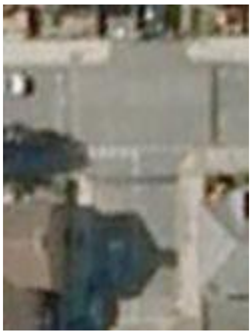


Figure 8-2 (Road intersection covered by shadow region)

The most notable feature of the Haar classifiers is the low false detection rates when compared to the methods edge based building detector used in previous chapters. This makes it very suitable for use in the vision based positioning system as the matcher finds it difficult cope with false positives.

The Haar Classifier took about 0.7 seconds to detect objects in a 640x480 pixel image. This time is based on an intel i7 2.4Ghz processor, running a python implementation. Only once classifier was evaluated at a point in time. At low altitudes the building classifier was used, but at higher altitudes the road intersection classifier was used for intersection detection.

Building Detection Results

A few of the building detection results for built up areas are displayed below

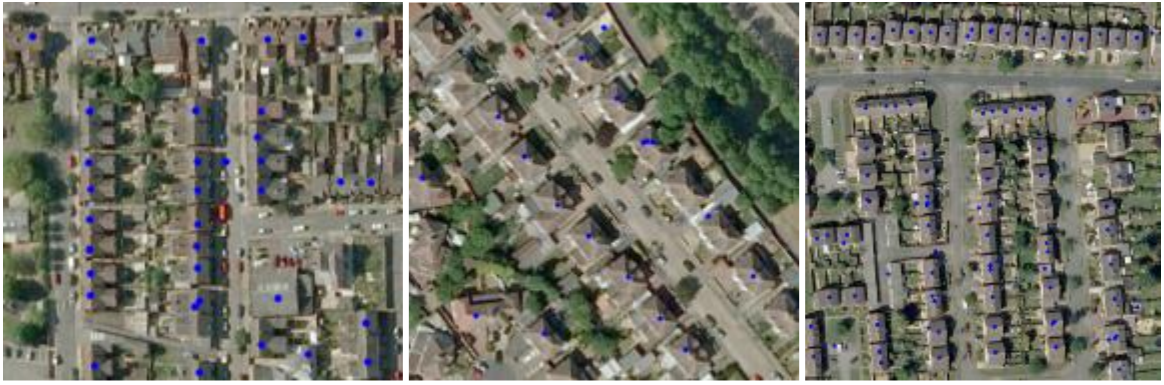


Figure 8-3 building detection results

It can be seen that in general the buildings are correctly detected in built up areas with only a few missed buildings. It can also be seen that there are very few false detections. The best building performance was found with builings that are simple square shapes as shown in the image below.

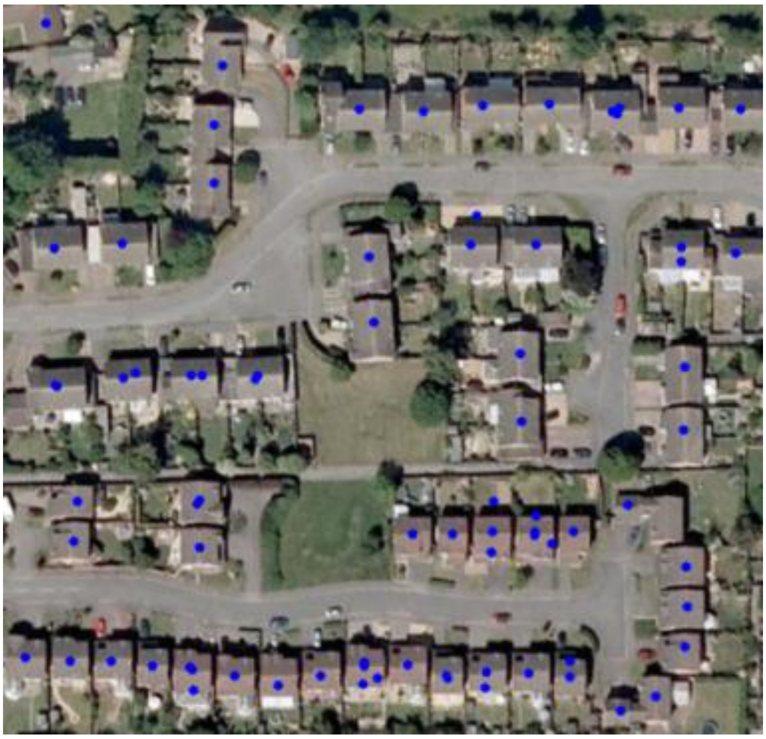


Figure 8-4 the best building detection results obtained for square shaped roofs

Very few false detection were obtained over regions with no buildings present as shown below for more rural areas. A problem encountered with rural areas is that some farm houses/ sheds are fairly unique and were not included in the training dataset. Because of this they are not correctly detected.

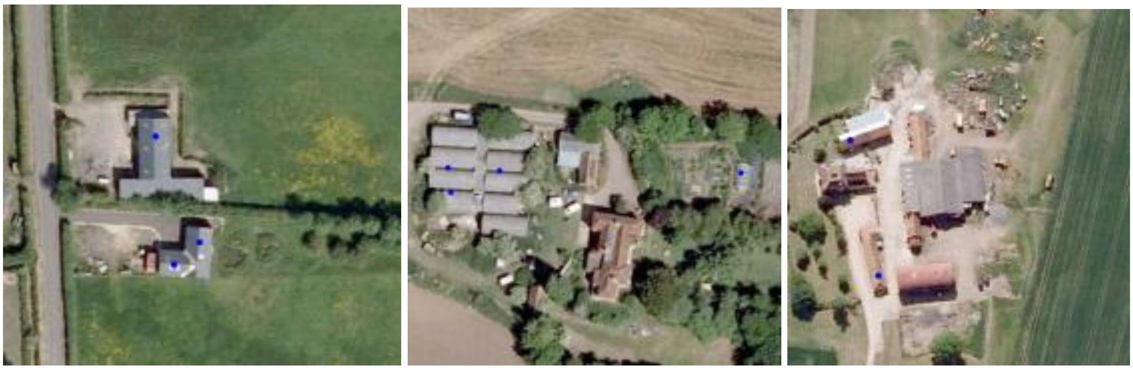


Figure 8-5 building detection in rural areas

Road Intersection Detection Results

A few of the road intersection detection results are displayed below. As mentioned previously the road detector was computed for sections of the flight that were at higher altitudes.

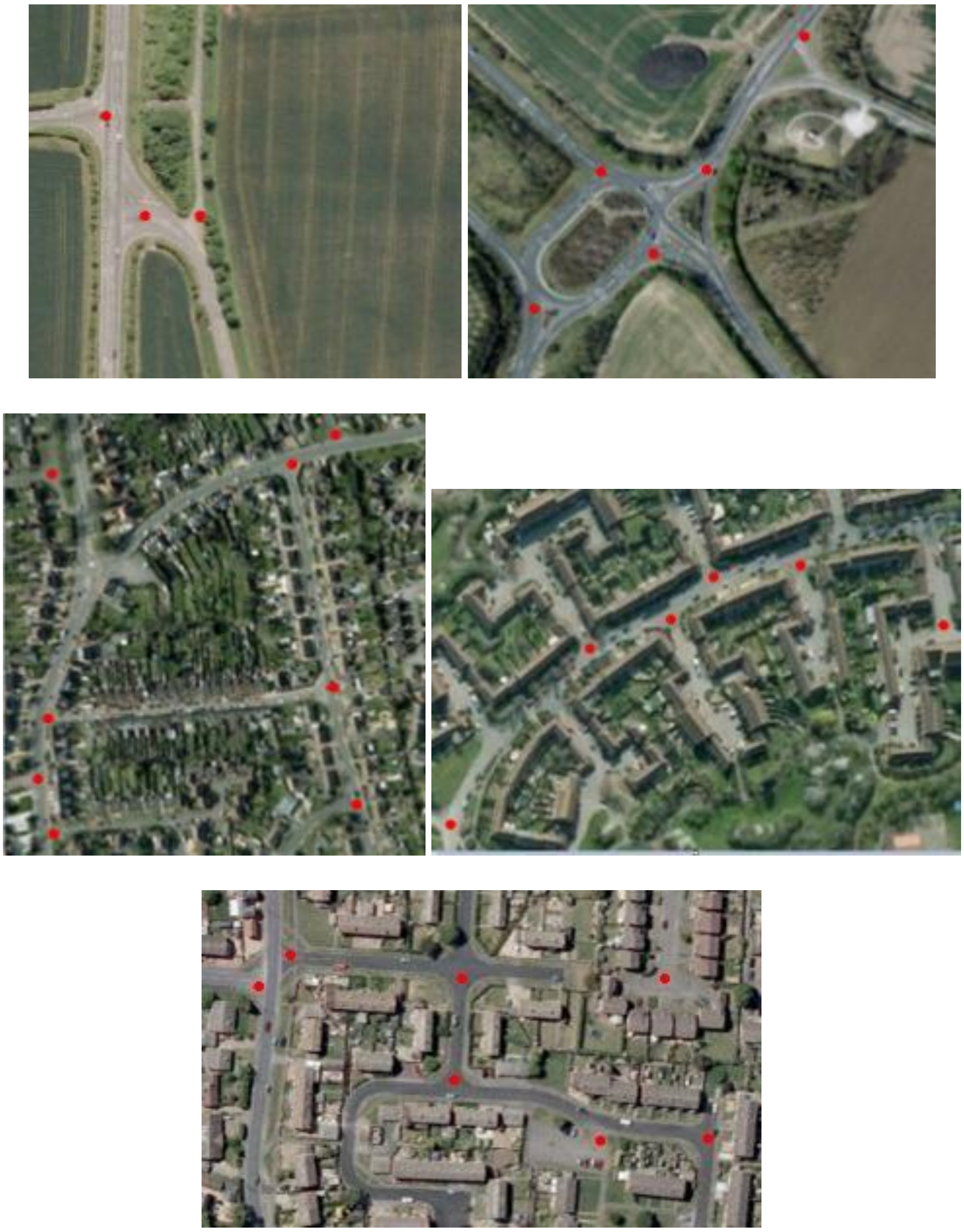


Figure 8-6 road intersection detection results

In general the road intersection performed better at higher altitudes as there was less noise in the images and the road regions stand out at higher altitudes. As mentioned before there is a large variety in the types of road intersections which make it more difficult to reliably detect the intersections.

Overall acceptable results were obtained for both intersection and building detection. Improvements can be made to both the classifiers in terms of adding more training samples to help the detector to handle the large variety of different road and building types. It was found that building detection requires a higher spatial resolution (lower altitude) for accurate detection, and the intersection detection requires a lower spatial resolution (higher altitude). These factors complement one another for later step of matching the landmarks because at lower altitudes there are generally not enough road intersections present in a given area for a match. Similarly at higher altitudes there is not enough spatial resolution to accurately detect buildings, but there are more road intersections present for position estimation.

9 Conclusions and Future Work

This thesis has presented and developed a few different methods for landmark detection and for visual positing systems. All of the methods proposed in this thesis have been shown to work in controlled tests, however the Haar based methods seemed to outperform all of the other methods in terms of robustness, and performance.

Using the edge based building detection methods provides an acceptable true building detection rate, particularly with FAST and Garbor filters, however the main problem is that there are still a high false detection rate. This will cause problems with the geolocation system that relies on the angle between the landmarks for accurate matching, if there are a large proportion of false positives it will be difficult to distinguish which are outlier landmarks.

The line based road intersection method has a unique advantage due to the fact that it extracts the road networks. These road networks can be used for other vision based navigation methods such as road following which could be investigated in the future. Also the linear feature detector can also be applied to detect other features such as hedge rows, railways etc. which could improve the navigations system when now flying over built up areas.

The main problems with the edge based building detection and line based road detection methods are the calculation time which are very slow and not suitable for real time applications.

The SURF key point detector was shown to operate in real time giving accurate position matches, however when the source and reference images were different the performance rapidly broke down. This method will require the reference database to either be updated regularly. The alternative is to also store key points at different seasons and conditions to establish a match for different source images; however this would require a much larger database that would take reduce the performance of the detector in terms of computational time.

The final method investigated what the use of the Haar classifier. The implementation in this thesis was able to create a rotation invariant detector as the cost of some computational speed. The haar classifier was able to detect a variety of different objects under different conditions which allows for a database that does not need to be regularly updated. A novel crowdsourcing method for fast training dataset creation was used which produced large datasets in little time. It was also noted that care must be taken to ensure the training images are of high quality to ensure a robust detector.

Overall this thesis has provided a robust method for landmark detection in aerial images that can not only be used in a vision based positioning system, but can also be used in other fields of research such as GIS systems. The detector used in this thesis can also easily be trained to detect other rigid objects in aerial images such as football fields, tennis courts, vehicles etc…

9.1.1 Future Work

In this thesis a functional system has been developed, however there is still room for improvements and future work to be conducted.

Speed Improvements

The work presented in this thesis has been developed and tested using Matlab, and Python. Implementing the system using a compile language such as C++ could result in a performance boost. However further work could look utilizing a Graphical Processing Unit (GPU) to offload some of the tasks from the processor which would result in a large performance gain. Future releases of OpenCV will have GPU support for the python library which will enable this to be investigated.

Furthermore, it was noticed that at higher altitudes, or when the aircraft is flying slowly the same object may be present in multiple frames. Tracking landmarks, using IMU data, or visual odometry once they have been initially detected, could allow the haar classifier to only look at new regions in an image, which could result in a further improvement in speed.

Larger training dataset

From testing it was found that the Haar classifier could not accurately deal a few less common variations of buildings and road intersections that were not included in the training dataset. For this reason it would be beneficial to include a larger more diverse training dataset for buildings and road intersections to help the system to detect a larger variety of objects.
Further testing and training could also be conducted to see how the classifiers deal under special conditions such as snow.

Vision Based Positioning System

The work covered in this thesis focused on reliably extracting landmarks from an aerial image. These landmarks will then be used for matching onto a database to estimate the location. As mentioned in the first chapter this thesis fits into a larger project of the vision based positioning system as a hole, so future work will look at brining all the individual pieces together such as matching, and visual odometry.

Using additional Landmarks

This thesis has only investigated detecting buildings and road intersections, future work could look at detecting other landmarks like warehouses, churches, tennis court, football fields, lakes and rivers. This will help improve the positioning system as certain landmarks occur less often so when one if detected, a position can be found fairly quickly.

Visual Positioning in rural areas

The main focus on this thesis was todetect landmarks in populated areas where buildings and roads are fairly common, however little has been done to solve the problem when no buildings and roads are present. Optical flow/ visual odometry systems can be used[1] when flying over such areas, however they are subject to drifting with time so they are not always suitable for long durations.

A possible solution to this problem when flying over regions with no roads/buildings would be to look at matching natural landmarks that do not change very often. These landmarks could be things such as small forest boundaries, although the boundaries would change slightly over time as new trees grow, they do not change very quickly. An example of such a natural landmark is shown below across a timespan of 4 years.

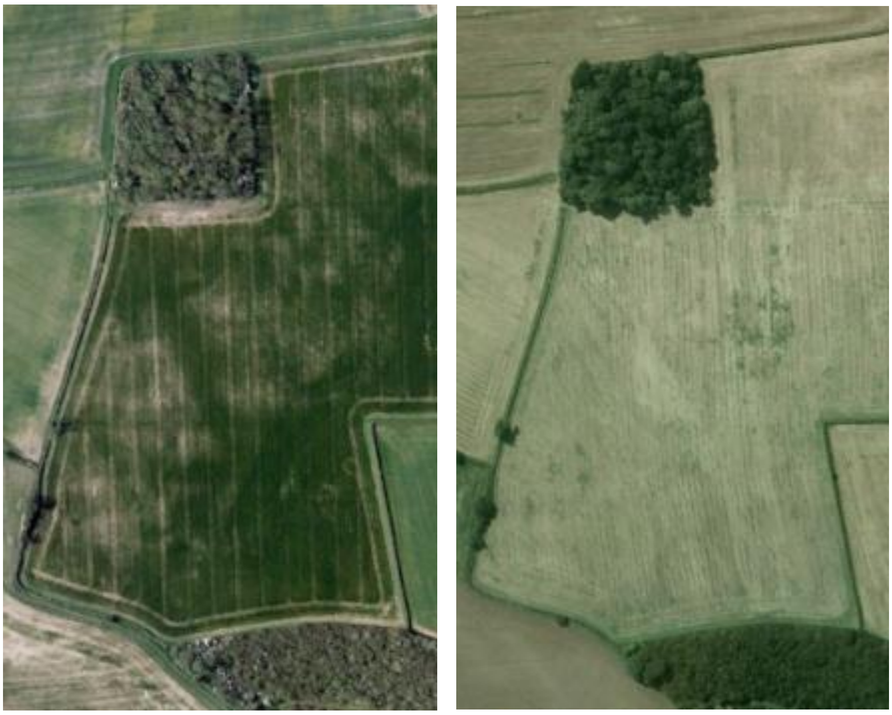


Figure 9-1 – two images of the same area showing how certain natural landmarks do not change over time (left taken in 2007, right taken in 2003)

As shown in the figure above, you can see how the small forest does not change very much with time, and would be suitable for investigating further for use as a natural landmark. Similar to small forests, other natural landmarks such as lakes, rivers would also be ideal, although less common. An important aspect to bear in mind with some natural landmarks is that they do change appearance over the course of a year (most trees lose their leaves, and change color) so the detector would have to deal with such changes.

Most of the United Kingdom is covered by farmland, and for this reason they can also be used for visual positioning. A typical farm/field will change many times throughout the course of a year as the new crops are planted/harvested. However as shown in the image below, the boundaries do not change very much. The image shows how a farmland changes from 1945 all the way to 2009. Some of the field boundaries do change as a farmer decides to expand, or divide the fields, however it can be seen that the majority of the field boundaries remain the same. This is more true for smaller differences in time. Interestingly it can also be seen that the small wooded areas have remained constant since 1945 which shows that they are a good potential landmark to use.

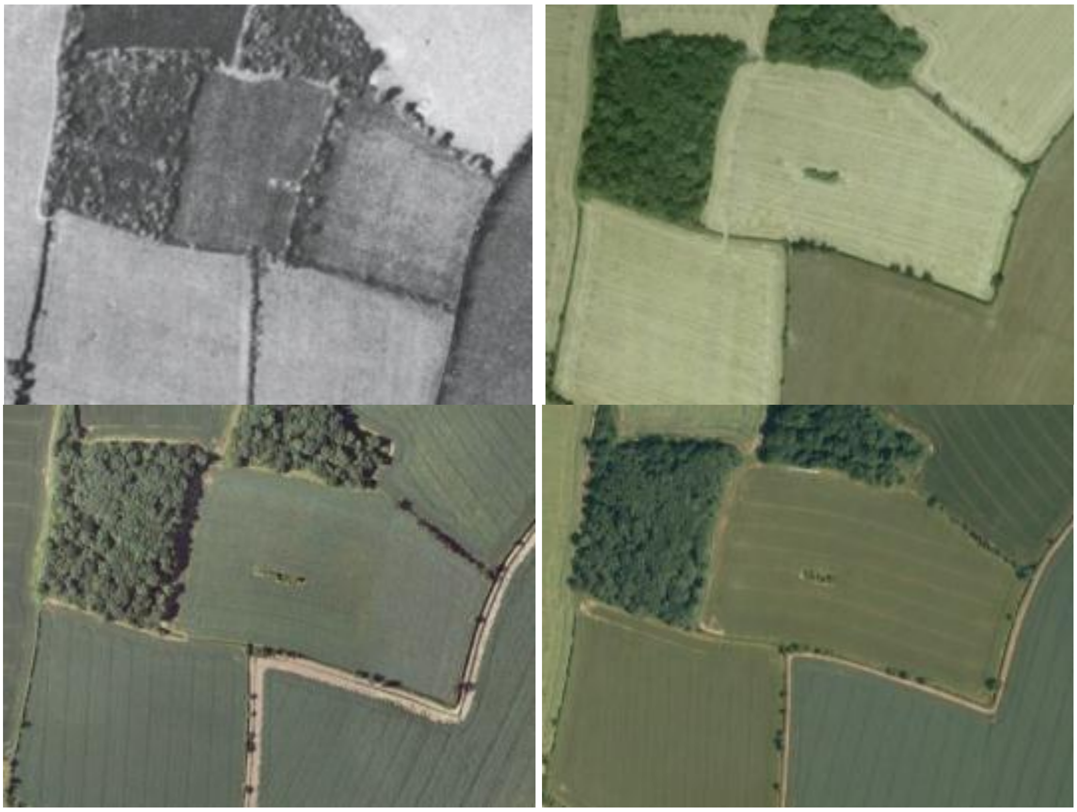


Figure 9-2 Aerial images shown how farmland changes from 1945(top left)-2009 (bottom right)

Using the field boundaries would allow a position to be easily matched by using the intersections between fields, this would require a simple edge detector algorithm to find all the edges, perform some post-processing to remove the noise. A result of the canny edge detector is shown below.

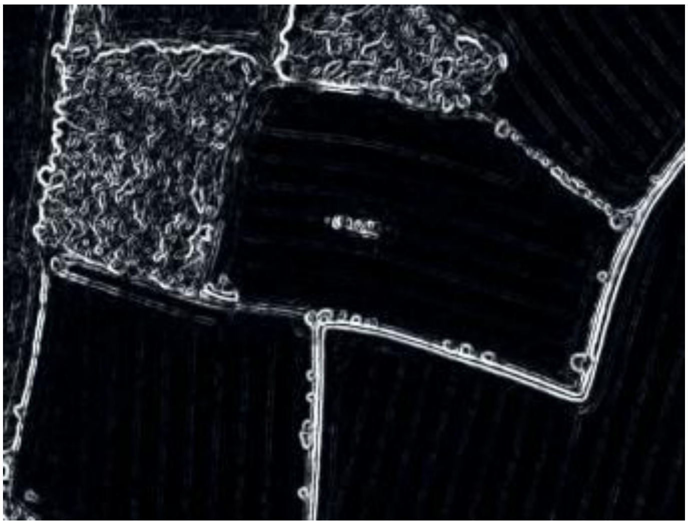


Figure 9-3 – Edge detection on rural region

As shown above the main boundaries of the fields can easily be extracted using edge detection techniques, however it is also possible to use segmentation, or a number of other methods to detect each field. Future work will have to find the most suitable method that can handle a variety of different environments and conditions.

Visual attention

Visual attention is a very interesting and somewhat unsolved research area in computer vision. This area looks at mimicking the human vision system that uses attention mechanisms to limit processing to only the important information relevant to the current task. For example, when you are reading this sentence, your brain/ eyes are focusing attention to the current word that you are reading, and dismiss all of the other visual information that is within your view. This way you can efficiently process only the information that you require. The human vision system does not always start of the top left corner and look at each region of a scene until it finds what it is looking for, if you walk into a room looking for a specific person, you will glance over all the faces in the room until you see the person you are looking for, your brain will automatically dismiss all of the other information or objects in the room such as chairs, tables etc. The first ideas on visual attention were proposed in phycology during the study of human visual perception back in the early 1980's [31].

In computer vision much research has been conducted in this area to avoid the computer algorithm processing each pixel in an image to find a given region/ object, as this can be computationally expensive and slow. There are many different visual attention models that are used in computer vision[32], however they are based on saliency maps, the architecture and an example of a saliency map is shown below.

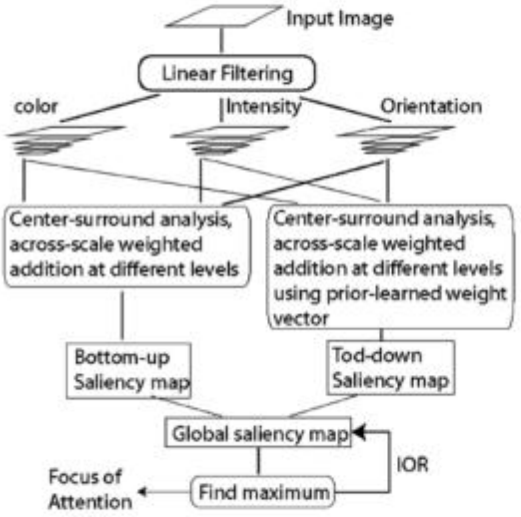

Input Image
Linear Filtering
color
Intensity
Orientation
Center-surround analysis, across-scale weighted addition at different levels
Center-surround analysis, across-scale weighted addition at different levels using prior-learned weight vector
Bottom-up Saliency map
Tod-down Saliency map
Global saliency map
IOR
Focus of Attention
Find maximum

Figure 9-4 Architecture of saliency map creation, further details can be found in[33]

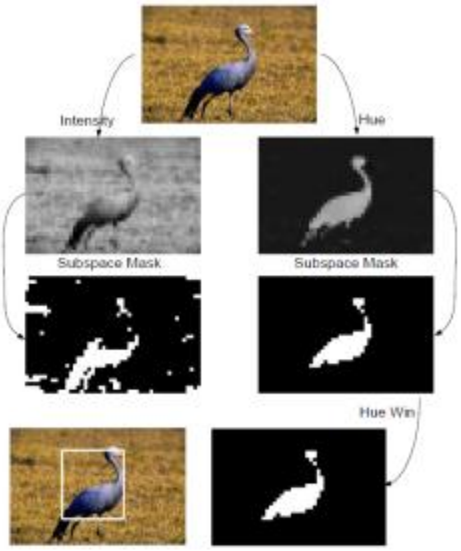


Figure 9-5 Example saliency map for a given image[34]

A saliency map is created for a given input image, based on some basic features/linear filtering methods that look at colour, intensity, and the orientation of the image. This saliency map can be used to focus the attention to certain regions of the image that have a higher probability of containing the object to be detected. Other techniques can also look at movement in the image. These linear filtering techniques are very fast and efficient. This linear filtering is based upon the object that you are looking for, or based on local contrast. The center surround analysis is then computed, which is based on finding the Gaussian pyramid at multiple scales, and then calculating the differences between each scale, resulting in strongest reposes at multiple scales being found. Best results are found when combining both the bottom-up and top-down saliency maps, however the details of which can be found in [33]. The basic principle is that the Bottom up saliency map simply shows the strongest responses of regions that stand out, or most salient regions, but the top down, will focus on the more relevant regions based on other information about the object to be found. To use this top down approach (visual search), offline learning is required so that the computer can recognize key features of the object to be found.

The use of visual attention algorithms quickly dismisses regions that probably do not contain the object to be removed, allowing the object detection algorithm to only look at smaller regions of the image. This could result in a much faster object detection system.

Furthermore visual attention, or other methods could be used to quickly distinguish a suitable classifier to use. For example if the aircraft is flying over rural regions, it is useless to run the building detection classifier as there are already no buildings to be found. It would be more feasible to try and detect more suitable landmarks like lakes, hedges that are more likely to be found in rural areas.

More details and a good overview of the latest advancements in visual attention can be found in a recent survey.[32] This will give a good understanding of this area and would be a good starting point for future work.

10 Bibliography

[1] G. Conte and P. Doherty, “Vision-Based Unmanned Aerial Vehicle Navigation Using Geo-Referenced Information,” *EURASIP Journal on Advances in Signal Processing*, vol. 2009, pp. 1-19, 2009.

[2] L. Wu and Y. Hu, “Vision-aided navigation for aircrafts based on road junction detection,” *2009 IEEE International Conference on Intelligent Computing and Intelligent Systems*, pp. 164-169, Nov. 2009.

[3] C. Steger, “An unbiased detector of curvilinear structures,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 2, pp. 113-125, 1998.

[4] K. Celik and A. Somani, “Mono-vision corner SLAM for indoor navigation,” *2008 IEEE International Conference on Electro/Information Technology*, pp. 343-348, May 2008.

[5] S. Ahrens, D. Levine, G. Andrews, and J. P. How, “Vision-based guidance and control of a hovering vehicle in unknown, GPS-denied environments,” *2009 IEEE International Conference on Robotics and Automation*, pp. 2643-2648, May 2009.

[6] J. Porway, K. Wang, B. Yao, and S. C. Zhu, “A hierarchical and contextual model for aerial image understanding,” in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, 2008, pp. 1–8.

[7] M. Izadi and P. Saeedi, “Automatic Building Detection in Aerial Images Using a Hierarchical Feature Based Image Segmentation,” *2010 20th International Conference on Pattern Recognition*, pp. 472-475, Aug. 2010.

[8] M. Awrangjeb, M. Ravanbakhsh, and C. S. Fraser, “Automatic Building Detection Using LIDAR Data and Multispectral Imagery,” *2010 International Conference on Digital Image Computing: Techniques and Applications*, pp. 45-51, Dec. 2010.

[9] H. Mayer, S. Hinz, U. Bacher, and E. Baltsavias, “A test of automatic road extraction approaches,” *International Archives of Photogrammetry, Remote Sensing, and Spatial Information Sciences*, vol. 36, no. 3, pp. 55–60, 2006.

[10] A. Grote, C. Heipke, F. Rottensteiner, and H. Meyer, *Road extraction in suburban areas by region-based road subgraph extraction and evaluation*. IEEE, 2009, pp. 1-6.

[11] P. Viola and M. J. Jones, “Robust Real-Time Face Detection,” *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137-154, May 2004.

[12] M.-level Uav, T. P. Breckon, S. E. Barnes, M. L. Eichner, and K. Wahren, “Autonomous Real-time Vehicle Detection from a Medium Level UAV (Haar Classifiers),” *Systems Research*, pp. 1-9.

[13] H. Bay, a Ess, T. Tuytelaars, and L. Vangool, “Speeded-Up Robust Features (SURF),” *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346-359, Jun. 2008.

[14] S. Juan, X. Qingsong, and Z. Jinghua, “A scene matching algorithm based on SURF feature,” *2010 International Conference on Image Analysis and Signal Processing*, pp. 434-437, 2010.

[15] D. Y. Gu, C. F. Zhu, J. Guo, S. X. Li, and H. X. Chang, “Vision-aided UAV navigation using GIS data,” in *Vehicular Electronics and Safety (ICVES), 2010 IEEE International Conference on*, 2010, pp. 78–82.

[16] G. Conte and P. Doherty, “An Integrated UAV Navigation System Based on Aerial Image Matching,” *2008 IEEE Aerospace Conference*, pp. 1-10, Mar. 2008.

[17] D. G. Lowe, “Distinctive Image Features from Scale-Invariant Keypoints,” *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91-110, Nov. 2004.

[18] M. Muja and D. G. Lowe, “Fast approximate nearest neighbors with automatic algorithm configuration,” in *Science*, 2009, vol. 340.

[19] C. Harris and M. Stephens, “A combined corner and edge detector,” in *Alvey vision conference*, 1988, vol. 15, p. 50.

[20] E. Rosten, R. Porter, and T. Drummond, “Faster and better: a machine learning approach to corner detection.,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 1, pp. 105-19, Jan. 2010.

[21] A. K. Jain, N. K. Ratha, and S. Lakshmanan, “Object detection using Gabor filters,” *Pattern Recognition*, vol. 30, no. 2, pp. 295–309, 1997.

[22] P. J. Green, a. H. Seheult, and B. W. Silverman, “Density Estimation for Statistics and Data Analysis.,” *Applied Statistics*, vol. 37, no. 1, p. 120, 1988.

[23] A. W. and E. W. R. Fisher, S. Perkins, “Hit and Miss Transform,” 2003. [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/HIPR2/hitmiss.htm. [Accessed: 29-Apr-2011].

[24] R. Lienhart and J. Maydt, “An extended set of Haar-like features for rapid object detection,” *Proceedings. International Conference on Image Processing*, vol. 1, p. I-900-I-903.

[25] “Amazon Mechanical Turk.” [Online]. Available: https://www.mturk.com/mturk/welcome. [Accessed: Oct-2011].

[26] A. Torralba, B. C. Russell, and J. Yuen, “LabelMe: Online Image Annotation and Applications,” *Proceedings of the IEEE*, vol. 98, no. 8, pp. 1467-1484, Aug. 2010.

[27] "OpenCV - Open Source Computer Vision Library." [Online]. Available: http://opencv.willowgarage.com/wiki/. [Accessed: Oct-2011].

[28] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face description with local binary patterns: application to face recognition.," *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, no. 12, pp. 2037-41, Dec. 2006.

[29] Y. Freund and R. Schapire, "A short introduction to boosting," *Journal-Japanese Society For Artificial*, vol. 14, no. 5, pp. 771-780, 1999.

[30] P. Li and I. Science, "Robust LogitBoost and Adaptive Base Class (ABC) LogitBoost," *Compute*, no. 2.

[31] A. M. T. and G. Gelade, "A feature integration theory of attention," *Cogn. Psychol.*, vol. 12, p. 970136, 1980.

[32] M. Begum and F. Karray, "Visual Attention for Robotic Cognition: A Survey," *IEEE Transactions on Autonomous Mental Development*, vol. 3, no. 1, pp. 92-105, Mar. 2011.

[33] S. Frintrop, *VOCUS: A visual attention system for object detection and goal-directed search*, vol. 3899. Springer-Verlag New York Inc, 2006.

[34] Y. Hu, D. Rajan, and L. T. Chia, "Robust subspace analysis for detecting visual attention regions in images," in *Proceedings of the 13th annual ACM international conference on Multimedia*, 2005, pp. 716–724.