Automatic Selection of Image Regions For Active Fixation

Heidar Hosseini

Department of Computer Science

Aberystwyth University

A thesis submitted for the degree of

Doctor of Philosophy

(PhD)

Acknowledgment

I would like to thank God for guiding and helping me make the right decisions throughout my PhD dissertation.

My special thanks to my mother Mrs Sajedeh Mosawi who never gave up believing in me as well as my uncle Fadil Mosawi for all his general support throughout my PhD period.

I wish to express my deepest sense of gratitude to my sincere wife Zainab and my children, Mohammed and Zahra, for their encouragement, patience and constructive ideas. They all made the house a proper place for a researcher.

I would also like to thank my grandmother Mrs Nashidah Mandalawi as well as my uncle Mr Kamel Mosawi for all their support, prayers and advices throughout my PhD period.

Profound thanks to Dr. Mark Neal and Dr. Fred Labrosse for their scholarly guidance and meaningful suggestions that were actually fundamental for the completion of my present research.

Their real learned support and endless encouragement have given me the strength to accomplish this work and to look forward to a brighter future academic life.

Finally, I would like to dedicate my work to my deceased father Mr Mohammad Hosseini whose soul and support played a pivotal role in my success and allowed me to achieve what I have.

Abstract

Using the properties of an image for the purpose of camera fixation is a difficult task in robotics. In this research, the selection of pixels for use in a minimization process is discussed and compared against other possible strategies. The Slice Selection Process (SSP) which uses pixel displacement graphs to select the most suitable image region for camera fixation purposes has also been introduced. Choosing which slice to use is based on counting every graph peak and selecting the slice with the fewest peaks for camera fixation. We also present hierarchical search strategies that aim to perform as well as an exhaustive search with an enormous reduction in pixel processing. The performance of both the hierarchical search and SSP methods will be assessed in both laboratory and real world environments. Moreover, the difference in performance of various lenses with different angle views was tested to observe the effect of the large field of view on the ultimate outcome. The laboratory experiments were performed by hanging the platform over printed images to observe the system performance on various views. Nevertheless, to escape the constant environmental conditions in the laboratory environment (i.e. light/wind), real world experiments were performed to test the system performance under varying conditions. Some investigations were also carried out to study the influential factors (i.e. edges, regions, colours) on the ultimate results.

Contents

Chapter 1

Introduction

Systems which permit camera stabilisation on platforms that are subject to cyclic motion is a difficult task in robotics. Using the visual input is highly dependent on the image's properties. For instance, stabilisation in an area which contains various visual similarities introduces issues with aliasing which may prevent successful re-stabilisation after a displacement of the camera/robot system.

Our research problem is to ask whether it's possible to use camera images to stabilise a moving platform in real time.

However, the research question that we wish to answer is:

Can appropriate regions for image-based stabilisation be selected empirically using pixel-wise comparison and simulated platform motion?

This would address the problems that are inherent in the visual stabilisation on those platforms which suffer from cyclic motions.

Research experiments were applied in kite flying and boat scenarios. In the flying scenario, the platform was suspended beneath a kite in order to take aerial photographs, whereas in the boat state the platform is attached vertically to the boat's side, pointing the camera towards the horizon level.

Figure 1.1 demonstrates the directions of the two cyclic motions where 1.1.a refers to a scenario of an attached platform to the boat's side and 1b refers to the cyclic motion direction of the kite flying state.

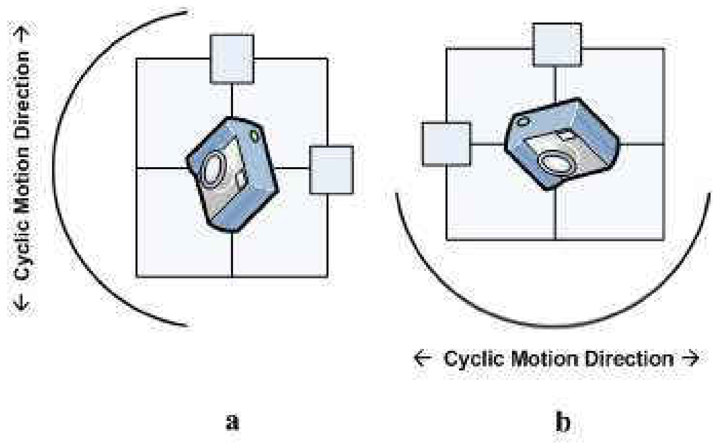


Figure 1.1

The designed platform uses a low computational overhead image processing technique to stabilise the camera in order to take accurately targeted images of the site being examined.

The ultimate design requires a wireless connection between the designed platform and the user's PC. To make the platform's design simple and light weight, we have built the platform with a Nano ITX PC which uses a lithium battery to operate.

The platform uses the camera images themselves as the only input to provide information about how the platform is moving with respect to the ground area being photographed. The aim is to keep the camera stabilised over a certain view and keep the user(s) updated with frequent stabilised images. The original view is determined after the appearance comparison technique is applied and is based on the result that the camera will be pointed back to the original view using the attached servos. We use a camera to provide us with our input data: frames extracted from the continuous image stream using a machine vision camera. The basic strategy takes two consecutive images and attempts to calculate the distance in terms of the number of pixels moved, and passes the output to the control system in order to adjust the platform's actuators to move the camera's centre of view back to its original location. The goal is to stabilise the camera to a single location, regardless of how far we move the camera (within the bounds of the actuator system and the field of view of the camera). This project is designed to reduce the need for external hardware such as gyros and high computational overhead techniques such as distortion removal by using low overhead techniques and processing at relatively high frame rates. In this research, we examine the properties of images and the error surfaces that they generate and their interaction with control systems with a view to stabilising the platform under examination as a test case for a more general solution to the stabilisation and orientation determination of actively controlled camera-bearing platforms. Related techniques will be reviewed to assist us in defining an acceptable technique to achieve our goals [93].

This methodology is divided into five major steps which start by capturing the reference image and storing it in the memory which then starts capturing images to process the appearance comparison against the stored reference. As a consequence of this, the result of the applied comparison will be passed for calibration and ultimately to the platform's servos to stabilise the camera back to origin. Figure 1.2 presents the complete steps which are followed within the designed methodology.

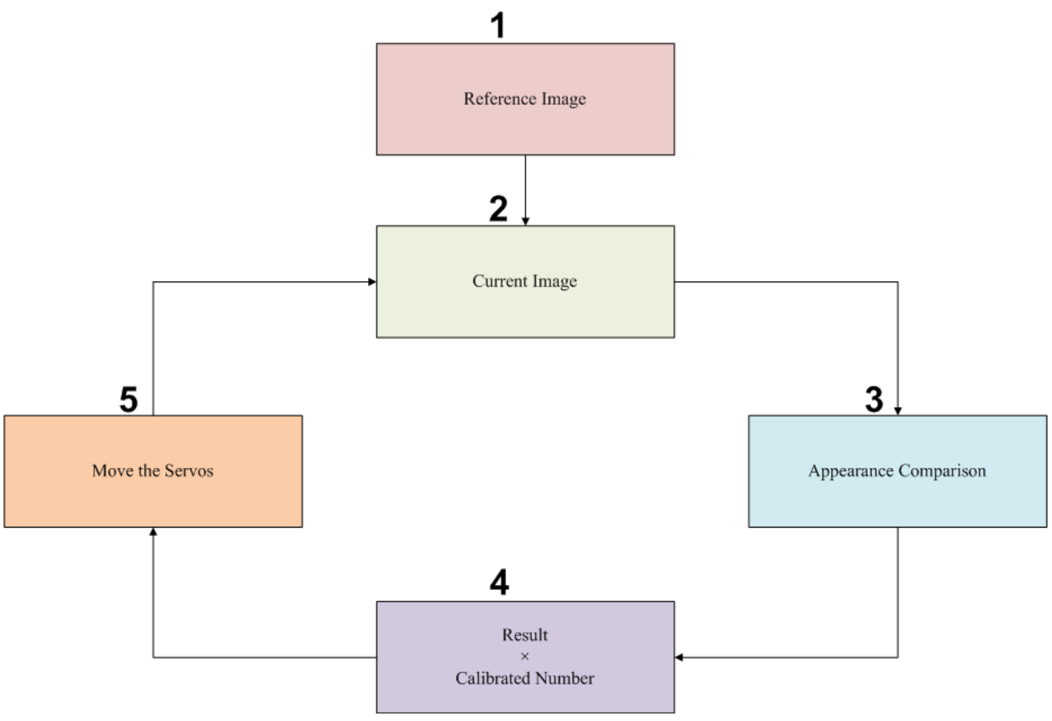


Figure 1.2 – Loop process

Step 1 (Reference Image) is applied once as the initial step but steps 2 to 5 are processed continuously within a closed loop.

The following chapters will explain the presented steps in more technical detail. Chapter 2 refers to the background and related studies within a similar area. Related studies will provide strong base on what image processing techniques are used and which ones are related to our question. In Chapter 3 we discuss the appearance comparison methods and compare the best technique amongst them all. It examines the different pixel selection methodologies to be used within our ultimate design. Chapter 3 will define the right method for selecting the appropriate regions for image-based stabilisation.

Chapter 4 presents the designed methods which will be used in our upcoming experiments. This examines how well the designed method answers the main research question.

Chapter 5 demonstrates the experimental results in the laboratory environment and in chapter 6 we illustrate the performance of the used stabilisation method in real world environments. The platform is tested in three different environments (Indoor, Outdoor and Sea) to show how it has challenged different factors to show its workability under various conditions. In Chapter 7 we present the conclusion and discuss the contribution of this research alongside the related literature. We also show that, it is possible to stabilise a platform using images and the developed SSP method allows this to be automated.

The two main contributions of this thesis are:

1) The use of pixel displacement graphs to choose image regions for active fixation. Pixel displacement graphs were used to help developed the Slice Selection Process (or SSP) which was tested in laboratory and real world conditions and shown to perform effectively under most circumstances.
2) The development of a hierarchical search technique which uses the selected image region to find the best matching position with minimal pixel processing. This was also tested under a range of conditions and was shown to be appropriate for real-time active fixation.

Chapter 2

Background and Related work

2.1 Preface

As explained in Chapter 1, the aim is is to address the problems that are inherent in the visual stabilisation on platforms suffering from cyclic motions, therefore some research on these major factors that are related to the visual stabilisation needs to be investigated.

The aim of this chapter is to address those related factors which are influential on the visual stabilisation technique. These factors are investigated from the input, tools and methods point of view, therefore, broadly speaking, we have categorised this chapter into two major divisions ("Input" and "Tools & Methods"). Most of the investigations in this chapter are targeting the robust visual tracking methods to track object(s) within the scene to achieve the most acceptable visual stabilisation possible [3], [19], [50], [104] and [116]. Those methodologies differ in terms of their design, prerequisites, outcome or the speed.

2.2 Input

The "Input" section was subcategorised as "Cameras" and "Colour Spaces". The listed tools/methods in each subcategory share the similar input scheme used in their work.

2.2.1 Cameras

The use of cameras for a range of purposes is intuitively attractive from an engineering perspective as they can provide apparently unambiguous signatures for many physical locations and can be used for the wide range of activities that robots undertake. The choice of camera to be used is dependent upon what the application is required to achieve. There are many cameras with various methods of performance but which to choose is the concern of any platform designer, as they will need to observe the internal functionality and the output of each to know which camera is the best to select. Broadly speaking, for machine vision, we need to use two types of cameras: the Progressive Area Scan or the Line Scan cameras [131]. Unlike the Interlaced cameras, with progressive area scan cameras, the whole picture is painted at once, which significantly decreases the flickering people perceive when watching TV. They repeat this process every sixteenth of a second, which makes it easier to cope with the speed of the human's vision or even faster than the eye can see [131]. The way that the "Interlaced" cameras operate is that they read the two discrete fields within the time interval in-between where the odd lines are first captured followed by the even lines. Ultimately, the process of merging will take place to put them together, producing a complete frame. In the case of a moving object or when the camera is on a moving conveyer, the possibility of receiving a blurred image increases. However, with interlaced scanning, the blurring issue is dealt with by outputting the frame as a doubled line, which will decrease this blurred view. Due to the ability of capturing the whole image at once, the industrial applications are using the Progressive Area Scan cameras more than before. The Progressive Area Scan is widely used in LCD computer monitors and in most HDTVs. The Line Scan Camera is another technology which is designed to capture a single line of pixels continuously. It is used to cover a 2D space, but as it captures a single line of pixels, the second dimension is obtained when the object is in motion or the camera does a manoeuvre on the object. One of the major advantages that the Line Scan has over the Area Scan is the ability to rotate around a cylindrical object and output all the surroundings as a 2D image space [131]. Choosing the type of output and the accessories that come with the camera is also as important as choosing which technology is to be used. In addition to that, capturing image from cameras with a wide-angle view solves many issues for applications that require capturing a large-angle view. Rotating imaging systems, fish-eye lenses [47] and the use of omnidirectional cameras [43] are three well-known approaches for the purpose of capturing a wide-angle view. Figure 2.1 demonstrates an example of a panoramic image from the garden's view. It clearly shows the captured 360 degree panoramic image, where position 0 is a randomly chosen position and the 360 refers to the complete 360 degree rotation back to position 0. This type of image enables the tracking/localisation to be much more reliable.

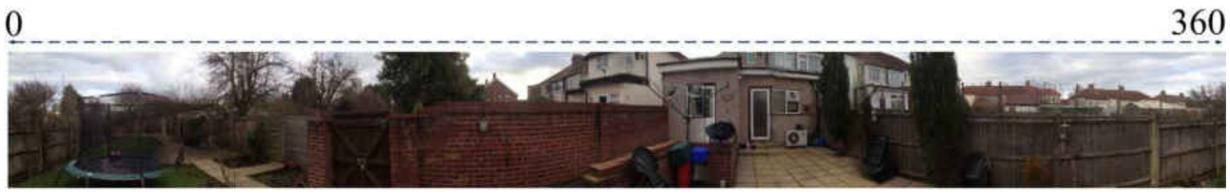


Figure 2.1 – 360 Degree Panoramic image. Position 0 refers to a randomly chosen position and 360 refers to the complete 360 rotation back to position 0.

Applications using omni-directional cameras to gain the best possible view, reap benefits in terms of disambiguation and local continuity of image variation, thus a larger angle of view is definitely a big advantage [43]. There are projects using panaoramic images for the purpose of tracking/localisation, however they mainly use similar technical methods to apply the image analysis process. That said, they differ in the way their ultimate goal is designed, for example, in [43], [47], [55], [72] and [87] the authors designed a method which uses panoramic images as input to estimate the robot's heading where the results will be compared against the results gained from the magnetic compass. In [43], the author used the Euclidean distance [34] method to define the minimal point and it seems that the whole images are interpreted with no extra sophisticated techniques (i.e. feature extractions). The way it operates is by regularly capturing panoramic images while the platform moves. Afterwards, every image is compared against the previously captured image which is achieved by unrotating both images and defining the lowest distance in the image's space. The author has carried out his experiments in both indoor and outdoor environments. Both of these environments were tested using both the visual and the magnetic compasses. The author has described that the data collected from the outdoor environment was not stable, which is mainly because of the uneven terrain that the robot was driving over. Therefore, it shows that the visual compass has performed acceptably according to the author's experiments, but it also suffers from possible repetitive colouring which the view may contain. The colouring repetitiveness decreases the possibility of capturing disctinctive features which would help in the tracking/localisation process. [30], [41], [108] and [111] demonstrate how the author has used the panoramic images as an input to apply a visual homing procedure. The term "Visual Homing" refers to the technique that the application uses to visually define its path to the original (or home) location. This process requires the application to have a reference image of the original (or home) location then compares the upcoming images with the stored reference.

In [111], the author used the Sony Aibo robot dog to play soccer fully autonomously, where it uses the panoramic images as an input and uses the SLAM [46] and [74] to extract the image's features and build a map of an unknown environment. Work in [30] and [108] have quite similar approaches. The author in [108] has used the Manhatan distance to define the distance between the images and that is achieved by comparing the initial image with the upcoming images, where the defined distance is mapped to the robot motion. However, as in most scenarios, the first stage is to unwrap the image and apply the appearance based method to determine the distance. Unlike the features extraction methods where we normally require prior knowledge of the features that we need to look for, the appearance based methods are independent of any prerequisites which make them more robust and reliable against any environmental changes. In [105], the author has shown that the insects navigate using only a retino-centric representation of the surrounding environment and no feature extractions are performed. The author has also applied the experiments using two different coloured spaces (RGB and CIE L*a*b*) and compared the results, which showed that there was a better rotation performance when using the CIE L*a*b* colour space. This encouraged the author to use the CIE L*a*b* for the whole experiment. However, the author argues that while the robot was driving towards the target, the accuracy of the robot's manouver towards the target decreases which indicates that this method is not a good solution for a “docking” station. In the previously listed papers {[43], [47], [55], [72] and [87]}, the panoramic imaging was used to track the position to either localise or stabilise the platform. The author of [71] has used two different methods for mobile tracking; one that uses the omnidirectional camera to retrieve panoramic images as the major input and the laser range-finder to estimate the range to the centroid of motion in the camera image. The second method uses the laser range-finder as a primary option and the omnidirectional camera as the secondary. The author has applied the experiments in dynamic and cluttered outdoor environments. The advantage of the laser can be the extraction of the 3D relative position of the blobs which may originate from the person, but it can only provide the range in a single plane. The system was built using the segway RMP platform, which is a good platform to use when a fast and stabilised condition is required. In [71], the author has used the Lukas Tomasi Kanade (KLT) [21] and [129] to track the features and compensate the egomotion by computing the bilinear pixel transformation between consecutive images. However, in order to start this process, the author split the raw omnidirectional image into a set of consecutive images with 45 degree spacing. The egomotion compensation and frame differencing is then applied to each prior image and the combined result is passed to a particle filter [36]. The author has used the EM clustering algorithm [96] to detect the peaks in the particle motion distribution. However, the way to detect the moving object is achieved by comparing the colour distribution histograms so when the correlation is found between two histograms, the matching between two images is determined. The author has argued that the visual input was only used to track the objects and the laser range-finder was just used to find the range of the moving objects. The author has also stated that the issues with the first used detection method in [71] was because the egomotion compensation did not take into account the actual environmental structure, when the nearby objects do affect the frame differencing. However, from the visual image processing point of view, the affect of having sudden obstacles on frame differencing differs depending on the object's size and the platform's moving speed.

2.2.2 Colour Spaces

Object tracking using the colours' properties is a current subject of research in many literatures and different colour spaces were used to retrieve different types of information for varying kinds of application. Each colour space is an abstract mathematical model which describes how the colours are represented as elements. The colour spaces of colour models are a way of representing the colours or reading the colour elements in different ways.

For instance, RGB and L*a*b* are two different colour spaces however, the RGB is one of the major colour spaces which some colour spaces are instantiated from. The RGB colour space is made up of three colour components (Red, Green and Blue). Unlike the RGB, the L*a*b* is made of two colour components (a* and b*) and the L* is used to specify the illumination of each colour.

These models are used by different applications and each model has some advantages and disadvantages, which makes them useable or non-useable for different sorts of purposes. For example, RGB is a widely used and well-known colour space which also enables the system to have a large variation on colour selections, but it suffers from the illumination issues because the colour's value may change if the surrounding light increases or decreases. However, this issue is resolved when there is another colour model such as L*a*b*, where the L* channel specifies the illumination of each colour.

These applications require either real time or non-real time processing and they were designed to process a specific methodology on the captured images or apply some manipulations to an image's properties. However, we will discuss the differences in the properties of different colour spaces in more detail. In RGB, each of the three channels has an assigned number that varies from 0 to 255 and a variation of these three makes a colour produced from the RGB Model. The RGB is the most common colour space which is used by a number of applications, or at least has been used amongst different colour spaces in many applications. The I1I2I3 [132] is one of the colour spaces that correspond to the conversion of the RGB colouring space. The I1 indicates the level of the grey tone axle of the RGB and the I2 plus indicates the I3 elements containing the colour information. The LSLM [132] colour space is also another linear transformation from the RGB colour space. Unlike the RGB, it is a mixture of four colours rather than three. The CMYK [132] colour model is also used in many colour printers and stands for cyan, magenta, yellow, and key (black). The K (or Key) in the four printing colours aligns the cyan, magenta and yellow printing plates with the key of the black plate. Unlike the additive colour models (i.e. RGB), in the CMYK model, the unsaturated case is when the white is dominating and fully saturated in the black. In such other models like RGB, the black is a demonstration of an absolute zero saturated situation with no colouring being added and the white is a case where the Red, Green and Blue are fully saturated. Like the RGB, the CMYK is a device dependant model. However, the conversion process from one model to another may not be an easy process and also may not result in an absolute accurate outcome. XYZ colour space was created by the International Commission on Illumination (CIE) in 1931. The quality of the colours is derived from the x and y and the Y is to measure the illumination of the specified colour.

The relative intensities of the primary colours (i.e. Red, Green and Blue in the RGB Colour space) are the tristimulus values. The X, Y and Z in the XYZ [132] model are its tristimulus values.

In the L*a*b*, the L* channel is to specify the illumination of the colour which in many cases helps to determine the actual colouring, i.e., whether the lighting had an influence on increasing or decreasing its brightness. The a* and b* channels are to specify the colour's properties where the a* specifies it as a Green or Magenta hue and the b* identifies the colour as a Blue or Yellow hue. It is one of these colour spaces that are normally used to distinguish the illumination from the colouring.

HSL is another colour space which also splits the colours' properties from the illumination. It stands for Hue, Saturation and Luminance. Luminance refers to brightness, where the Hue and the Saturation are related to the colours' properties. The HSL and HSV [132] are a transformation process from the RGB colour space and the actual H and S are related to the sort of RGB space they belong to. HSI (Hue, Saturation and Intensity) is another colour space which is widely used in the computer vision applications. It is a well-known colour model from the image processing point of view, as it represents the colours in the same way that the human eye can see them. To find the values for all HSL, HSV and HSI colour spaces, there is a well-defined process of how to convert RGB to one of these colour spaces. Y'UV [132] is another colour space which was created from the RGB source. There are many other colour spaces which also have similar properties as the Y'UV, which takes human's perceptions into account when image processing techniques are involved. However, the reason for using this colour space is to cope with the analogue or digital televisions and the photography equipment of those that are implemented using the Y'UV standards. Y' is for the luma which is used as a standalone component in black and white televisions. The U and V are for the colour components but they are added separately to allow the black and white televisions to receive colour pictures, but still display them in the black and white format. However, there are some other colour spaces which are based on the way the humans react to the colours. For instance, the YIQ [132] uses the knowledge of the colour response characteristic in the human eye. The human's perception is intended to be more reactive to the changes in the orange to blue range and less in the purple to green range. This colour space considers these two cases and divides them into two components, where the I refers to the changes for the orange to blue range and the Q refers to the changes for the green to purple. As the human perception is more reactive to the “I”, more bandwidth will be required than for the “Q” component. Different applications are used to demonstrate the difference in outcome when different colour spaces are used. The “Visual Homing: a purely appearance-based approach” [108] is one of those approaches which compares the difference in performance. For the homing approach visual analysis is employed when different colour spaces are used. It uses panoramic images that are processed and the result is to be sent as feedback to the robot’s controller part. Usually, the comparison is made between two types of colour spaces. One is the most ordinary colour spaces, such as RGB and the other model applies more intelligent algorithms. For instance, in the “Visual Homing”, the comparison of the process was between the RGB and the CIE L*a*b* colour spaces. The comparison must always be made between the same image using a similar distance function and other facilities. In the “Visual Homing” approach, the robot’s rotation was achieved more accurately when the CIE L*a*b* colour space was used. The main reason for this was because the colours are eliminated from the brightness, which is caused by the surrounding lights. For instance, if we have a red ball next to an orange wall, then increasing the environment’s light could decrease the saturation of the ball’s colour, which may result in having the ball closer to the wall from the colour point of view. These sorts of issues are usually dealt with by using these colour spaces, which eliminate the brightness from the colours’ components.

The colour Histogram demonstrates the colour distribution across the whole of the image’s space. In digital images, the image’s histogram can be a fixed list of a range of certain pixel values or even an individual pixel. Therefore, by iterating through the pixel values, we would then be able to locate each of the pixels on a specific range value. In [128] and [98] authors demonstrated how the use of colour histograms is useful within the tracking scenarios. Some systems, such as [98], are not designed for tracking purposes but they have used similar techniques to assist the driver with detecting moving objects while the person is driving the vehicle. In [128], the colour histogram beside the gradient orientation histogram was used to provide the colour and the contour representation of the objects. The system [128] was designed to detect the objects with high discriminative properties to be used for the tracking purpose and also to be more robust against the conditional changes (i.e. light). The author of [128] has used HC in RGB and HOG on gray image data to develop the combined feature set, which was named HOGC. To make the system more robust to the rotation and deformation, the author has used the RGB colour space. The SIFT in techniques [78] and [84] was also used to make the system robust against the possible scaling. The object tracking was performed with an exhaustive search method in the candidate area [128]. However, the candidate area was defined by the use of the Particle filter [36] technique, which is widely used to represent the posterior state of an object's movement using a set of random variables.

2.3 Tools & Methods

As earlier explained, the aim is to address the problems that are inherent in the visual stabilisation on a platform suffering from cyclic motion. As a consequence of this, several related research studies have investigated relevant tools and methods. To present some of the most used tools and methods which were used for the tracking purpose, we have subcategorised this section into Probability Distribution, Edges and Segmentations, Contours, Features, and Adaptive Methods & Filters, although, the listed tools/methods in each subcategory share the similar methodology used in their tracking systems.

2.3.1 Probability distribution

Several literture studies like [15], [16], [17], [18], [42], [51], [59], [60], [73] and [103] have used the probability distribution techniques for tracking purposes. The content of the "Probability distribution" section is subcategoriesd as Gaussian Distribution, Condensation Tracking, Mean-shift, Covariance Tracker and Markov Chain Monte Carlo. However, each subcategory refers to one or more tools or methods used for tracking purposes.

2.3.1.1 Gaussian Distribution

The Normal distribution has been used in several literature studies on tracking methods, such as {[51] and [103]}. The Normal distribution was first discovered by Abraham De Moivre [51]. Normal distribution is also called Laplacian or Gaussian distribution. The Normal or Gaussian distribution is a probability distribution method that is used to gather data around a single mean value. It is completely characterised by the mean and the standard deviation and so, if we do have these two values, we will be able to define the proportional distribution values of the Gaussian distribution. Knowing the mean and standard deviation then means that we do not require the values of the individual pieces. To understand the Gaussian distribution by imagining that we have a vertically standing stick and dropping a ball on top of it, then we should assume there is a 50% equal chance of having the ball being directed to either the left or right of the stick. If we make a big triangle out of these sticks (see Figure 2.2) and drop many balls from the top, then the way the balls are distributed around the bottom part of the triangle is called a Normal (or Gaussian) distribution. Certainly, there is a higher possibility of dropping balls within the closest area from the middle of the stick or the mean value in the real case scenarios. Nevertheless, if the situation suits the normality where the values would need to be within a certain area then this method works well [51].

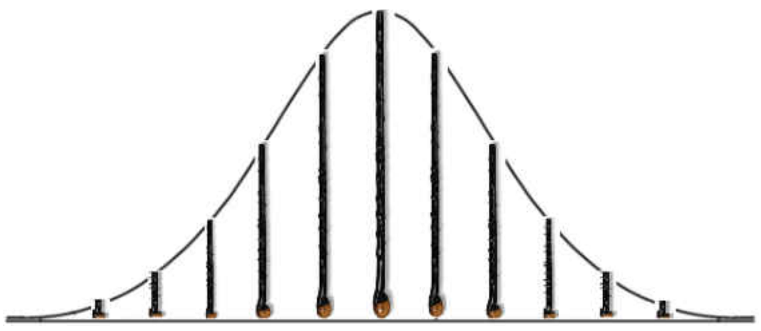


Figure 2.2 – Gaussian distribution [51]

Gaussians are generated by one of the many available techniques (i.e. particle filter), which would then be used for the tracking purpose.

In [103], the author has modelled each pixel as a mixture of Gaussians and the models are updated frequently. However, these models are evaluated to determine whether each location of the image is part of the background's model or not. Pixels which do not fit the background model are considered to be part of the foreground objects but, there is also a possibility that one Gaussian may include them when proper evidence is provided. These foreground pixels are grouped using connected components. With this procedure, the author could design a method to segment the background and recognise the moving objects within the scene. As the models are updated frequently, this system is capable of dealing with the long-time background or the lighting changes. Most researchers have stopped working with long-time tracking because there is no need for a frequent model updating process. However, when a new pixel is considered as part of the background's model, it will not destroy the existing model, but instead the existing model updates the currently used background's model. The initiation of messages of connected components between consequent captured frames is achieved with the linearly predictive multiple hypotheses tracking method, which fits in both position and size.

2.3.1.2 Condensation

The condensation algorithm (Conditional Density Propagation) is a well-known probabilistic algorithm that is used for detecting, localising and tracking the object's contour within a cluttered environment. This means that not every pixel is being processed but instead, pixels are chosen randomly which results in a huge reduction in the processing time. With condensation, we are able to define which parts or pixels make up the object's contour. The algorithm was described in detail by Michael Isard and Andrew Blake in [16]. More papers like [15], [17], [18], [42], [59], [60] and [73] were involved in using the Condensation algorithm for tracking purposes. However, the Condensation algorithm originated from the Particle Filter which uses a set of samples (or particles) to represent the propagation of arbitrary probability densities over time [42]. However, the author of [16] argues that the Condensation algorithm benefits from the ability to manage multiple hypotheses and provides a more general probability representation, making it a suitable algorithm to deal with the non-linear and non-Gaussian distribution. As tracking the "curve" is a big challenge with the available methodologies, the author of [16] demonstrated how the Condensation algorithm was used for curve tracking. However, to detect the curve's shape, the author used a well-known B-Spline curve detector method which is widely used in different literatures for the curve detection technique [80]. The procedure was to detect those curvy shape foreground objects. Nevertheless, as in many scenarios, the challenging part is to disallow some of the background scenes to be added to the foreground's contour, due to their shape or colour similarities. The accuracy of the object's detection will have a major effect on the tracking process in the subsequent frames. In the Condensation algorithm, the propagation process is to detect the foreground achieved within a time discrete "t", where two sets of values are known to the algorithm.

2.3.1.3 Mean-shift

Mean-Shift is a non-parametric feature space analysis technique which was first presented by Fukunaga and Hostetler in 1975 [45]. It is also called the Mode Seeking [124] algorithm. The "Mode" in statistics is the value which occurs most frequently. Mean shift counts the feature space as an empirical probability density function. In image processing scenarios, the input is a set of points (or pixels) where the mean shift algorithm will consider them as a set of samples from the underlying probability density function. If the dense region(s) do exist, they would correspond to the local maxima of the probability density function. Mean-Shift defines a window around each of the data points and computes the mean of that point. The method is designed to shift the window's central to the mean and repeats the process until it converges. Therefore, the Mean-Shift is considered as a Gradient Ascent method where the window is always heading towards the mean value. In [7], [29], [49], [54] and [88], authors have used the Mean-Shift algorithm for the purpose of the tracking procedure. In [54], the author has presented the development process of an aerial tracking system, where a helicopter is used to lift a platform which uses a camera to track the ground's target. The scenario begins by flying the platform over the target's region and then the operator on the ground station selects the target using the live broadcast. The tracking part was designed to use a modified version of the original Mean-Shift algorithm where the image's gradient is used for the purpose of target tracking. Like many similar applications, of the possible issues that the author has discussed in [54], the camera passes over a view too quickly which results in reacquiring different targets. In [29], tracking was achieved from a moving camera, which is a much harder procedure than being on a fixed camera. The author used the Mean-Shift algorithm as its central computation module and used the Bhattacharyya coefficient metric [64] to determine the dissimilarities between the model and the candidate. The similarities are measured by taking into account the probability of the classification error where it is directly related to the similarities of the two distributions, meaning the larger the probability error is, the more similar the two distributions are. Therefore, based on Mean-Shift iterations, the new target location is determined. As in many scenarios, the detected objects on the initial captured frame will be frequently updated for any possible changes in its colour's attributes. To achieve a better strength against the scale changes, the author modifies the kernel's radius by ±10% to increase the tracking robustness. In [49] the author has demonstrated the use of an optimised version of the SSD-Like measure using the Newton-Style iteration for the Mean-Shift tracking algorithm. The Newton-Style method was designed to find a better approximation to the root. Therefore, it can make fewer assumptions than the technique that is usually used in a Mean-Shift iteration, which makes it a much faster process. However, as the author of [49] also argues, one of the major disadvantages of using kernels is the rotation possibility and so if we wish to recover from any problems which may arise from this issue, we need to think about designing rotating kernels that check for any rotation possibilities. In [49], multiple kernels were used to increase the measurement space and the author has argued that the SSD measure extends naturally to multiple kernels. The scenario in [7] differs slightly in the way the objects are recognised, where the author has introduced the use of classifiers for the purpose of distinguishing between the objects and the background. The author has used an ensemble of weak classifiers into a strong classifier using the AdaBoost algorithm, which is an adaptive method used beside other learning algorithms to improve their performance. In [7], the strong classifier will be used to label the pixels and to classify each pixel as either belonging to an object or a background. This process is achieved by designing a feature vector for every pixel that belongs to the background. The iteration process will provide a confidence map which signifies peaks over those objects that do not belong to any of the background feature's vector(s). The Mean-Shift algorithm will then be used to find those peaks. The Ensemble Tracking in [7] will work continuously to update the stored classifiers in order to separate the foreground from the background and find peaks on the map believed to be the locations of the objects.

Work in [88] also used the Mean-Shift technique to track the target(s) however, the coarse-to-fine search was implemented in the simultaneous localization and mapping (SLAM) system [46] and [74]. Nevertheless, as it is well-known, the larger patch is much more residual to the motion blur but more expensive to compute but the smaller patches are faster to compute but they are not as residual to the motion blur as the larger patches are. The coarse-to-fine search implements a hierarchy of the strategy of searching from large size to small size patches and that is achieved by first searching for the large features with a large search radius and then searching for small size features using a small search radius. This makes the coarse-to-fine approach fast and robust to many possible changes. In [88], the author has presented how the coarse-to-fine approach was used to track the part-based model by employing spring systems to set the relationship amongst the parts. However, every part in the targeted scene has multiple features, where each of those features is assigned with an independent tracker and the features within every part are connected using a spring system. Figure 2.3 demonstrates how the multiple trackers are connected using a spring system and 4(c) presents how the coarse-to-fine process operates like a pyramid searching procedure by looking for a large feature and then starting its search for small features to enhance the accuracy level even further. For the tracking procedure, the Mean-Shift technique is used to associate the features between consecutive frames [88].

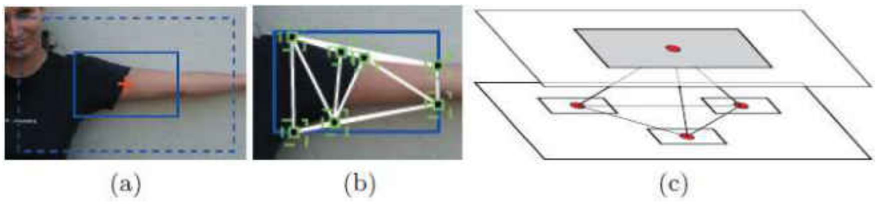


Figure 2.3 - Spring system connects multiple trackers [88]

Camshift [1] (or Continuously Adaptive Mean Shift) is another popular way of using the Mean-Shift technique, but it has the ability to change the window's size when it arrives to convergence. The Camshift technique is intended to be used for head and face tracking for a perceptual user interface. The main difference between the Mean-Shift and the Camshift algorithms is that the Camshift continuously uses adaptive probability distributions, while the Mean-Shift only uses the static distributions, which restricts it from being updated unless a significant change occurs in the colour or the shape of the object(s) [1]. However, as the Camshift does not update the static distributions, it will use the spatial moments to track the distribution's Mode. Camshift uses a one dimensional colour histogram with the use of a HSV channel. In [1], the author has extended the use of this technique to be able to track in an arbitrary number and type of feature space. To increase the speed of performance, in [1], the author only used the "hue" channel but difficulties arise when in some scenarios the hue alone cannot easily distinguish the foreground from the background.

2.3.1.4 Covariance Tracker

The use of the covariance matrix of image features for the purpose of object tracking has been widely used in recent research [5], [40], [63] and [121]. The methodology works by extracting various features (e.g. location, intensity, colour, gradient) and the features are represented by the covariance matrix in that region. The similarity between two covariances is measured on Reimannian manifolds [121]. To estimate the new object's location on the new frame, the new covariance is compared against the reference covariance. To improve the tracker's quality instead of having a full scan of all possible locations, for example in [5], the author has proposed the use of different techniques like Local Search (LS), Mean Shift Optimisation (MS-C) or Gradient Descent Optimisation (GD) to speed up the tracking process. However, as the author also argues, the computational efficiency of a tracking algorithm is as important as its performance. Figure 2.4 from [40] presents the flow diagram of the covariance tracker, where it starts by extracting the features from the input frames then constructs the covariance matrix. Consequently, the covariances will be compared to define the ones with a minimum distance in between them to determine the new object's location.

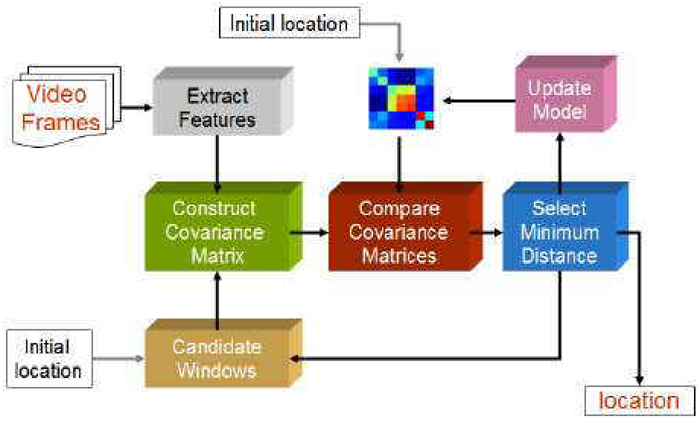


Figure 2.4 - Covariance tracker flow diagram [40]

2.3.1.5 *Markov Chain Monte Carlo*

Within the used tracking techniques, Markov also used the eye tracking procedure in [10] and [58] where the eye is detected and tracked using the human's face colour and geometrical features. The skin is detected using the skin colour detector. Eye tracking was used for many purposes, for example, in [10] the author decided to use the Markov based method which first decides whether the eye is closed or open and then uses the Markov chain to model the temporal evolution to determine the subject's gaze. However, in many scenarios, before processing the temporal evolution (using the Markov chain) some pre-processes are required to allocate the eye position. In [10], the author has used four points on the nostrils and eyebrows to detect and track the eye's location where the tracking operator was made robust to scaling and transalation which makes the eyes trackable, even when the face makes a fast movement. However, it is still possible for the system to lose the track due to speed or an occlusion. The tracking methods based on the Markov chain were also used for multi camera's monitoring projects [58]. The use of non-overlapping cameras is much more challenging, as the object can disappear from one camera and re-appear in another one with a different appearance. In [58], people's movements are monitored, the path from one camera is monitored in a probabistic manner (Markov Chain Monte Carlo (MCMC)) and the maximum posterior is determined using the schotastic transition model. For the detection process, the author of [58] used the background subtraction and a simple blob to detect the walking human's body. The paths and the movement objects are classified according to their depature and arrival locations within the entire topological map of the connected cameras [58]. However, MCMC is a probabilistic algorithm which was designed based on the Markov Chain. The Monte Carlo technique was created after the paper [20]. The technique started to grow with the development of computing performance which is now widely used in many literatures [109]. It is also important to know that, the more steps the function takes, the more improvement there will be to the quality of the samples but, we also have to recognise that it is a difficult process to determine the number of required steps to reach the desired range with an acceptable error. More papers such as [33], [67] and [68] have used the MCMC technique for different purposes, for example, in [33] the author used the MCMC technique to represent the uncertainty of localising the robot platform. However, in [67] and [68], the scenario is specially designed for the tracking purpose. The author of [68] introduces a modified particle filter method which use the Markov Chain Monte Carlo (MCMC) algorithm called hybrid Monte Carlo

(HMC), which is designed to determine the posterior in a high dimensional state space. With HMC, each particle produces a Markov Chain which follows the gradient of the posterior to the large distance, making it much faster than the conventional used particle filter [68]. More strategies, such as the Annealing Technique, are also used to allow MCMC to move between multiple peaks. However, as the author also argues, with Basyan methods where the samples are distributed randomly, care must be taken as some locations may get higher samples (or particles) during the distributions process than others which can lead to a poor performance. However, the HMC samples are generated and distributed in the best possible way according to the posterior, so when the algorithm follows the gradient to the posterior, it follows a path based on an effective distribution. As in [67], scenarios where the tracking is based on non-linear or non-Gaussian estimation [48] and [53], the sequential Monte Carlo will be an acceptable solution. We should also be aware that the more targets we track the more computational load increases, especially if the number of targets is unknown which also means the distribution of particles varies during the tracking procedure. Therefore, as the author of [67] also argues, we would need to minimize the number of particles to keep the speed of performance reliable and to allow the system to increase or decrease the particles whenever the number of targets contrasts.

2.3.2 Edges and Segmentations

Research studies such as [11], [23], [24], [35], [52], [56], [57], [90] and [101] have used the edge detection or the segmentation techniques for tracking purposes. The content of this section is subcategorised as Edge Tracking, Egomotion Compensation, Graph Cut, Background Segmentation and Template Matching. However, each subcategory refers to one or more tools or methods used for tracking purposes.

2.3.2.1 Edge Tracking

The process of identifying the ultimate point of the continuation of an object, feature, and colour on the image is declared as the Edge Identification process. Some research studies such as [11], [23], [24], [35], [52], [56], [57], [90] and [101], have carried out some investigations on the edge detection technique, while other research studies such as [91], [102] and [105] have focused their investigations more into edge tracking for the purpose of the navigation process. Edges are the major changes in the intensities and more variation means more edges occur. However, these changes are either sudden or caused by smooth variations. Also, it is either a step change to a different intensity level or a temporary change to a different value which returns to the original value after a short distance. The first case is referred to as "step discontinues" and the second is "line discontinues". However, because of the smoothing process that is applied by the sensing devices, these two edges are rarely possible which means that the "step" becomes "ramp" and "roof" will replace the "line" edges. Figure 2.5 is a graph representation of these changes to the intensity level.

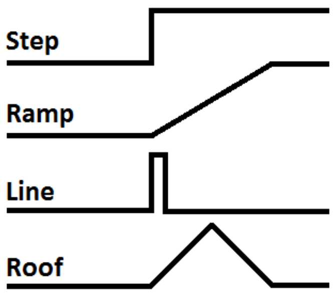


Figure 2.5 – Changes to the intensity level

The importance of the edges and its intensity variations may also indicate a sign of an object or feature detection. Sometimes, the edges do not have a step change in their intensity level but they will gradually change from one value to another. The operators also need to be designed for these sorts of cases and a shadow around an object is a good example of these types of edges. Real edges do have a high noise level and how the operator is programmed will detect whether this noise is an edge or something like a shadow of an object. The operators are designed to work over the 2D surfaces and identify the edges using the pre-programmed techniques and methodologies. Not every operator will suit all samples; this depends on the structure of the image. The edge orientation is how the operator was designed to be sensitive to a certain direction on the image. Some are designed to look for edges on the vertical, horizontal or diagonal edges.

2.3.2.2 Egomotion Compensation

In [47], [70] and [106], the authors have used the egomotion compensation methodology to detect the moving object from a mobile robot. Egomotion compensation is the technique that is used to determine the object's motion from two or more images using image processing techniques [106]. Nevertheless, in all mobile tracking applications, there are two sorts of motions that need to be counted. The first is the object's motions that need to be tracked and the second is the camera's motion which is mostly placed on a mobile robot. These two motions are technically merged, making a single complete motion from the camera's point of view. These sort of applications are different in the way they process the consecutive images to distinguish the two motions to determine either the main camera's motion or the tracked object(s) motion(s). In [106] the author has used two different methods to distinguish between the two motions. For the object tracking, the author has used the probabilistic approach where the Adaptive Particle Filter [36] and EM algorithm [96] were used. The author of [106] has applied the test of the algorithm on three different platforms (robotic helicopter, Segway RMP and Pioneer2 AT). Using the camera over these three different platforms gives three different problems that need to be solved. The Forward/Backward motion is the camera's only activities when it is mounted over a mobile robot. However, the camera's motion on the robotic helicopter is the pan/tilt movement which creates a different difficulty to be resolved than the scenario with the mobile robot (i.e. Pioneer2 AT). Since the Segway RMP uses two wheels and stabilizes using special techniques, which makes it act like an inverted pendulum, it differs from the Pioneer2 AT platform where the pan/tilt movement also needs to be activated beside the Forward/Backward manouver. The author of [106] has studied three different models (affine model, bilinear model and pseudo-perspective models). The affine model is mainly suitable for a linear transformation which can mostly be useful when the platform makes a slow motion. However, if the camera's movement is fast then we will encounter a high non-linear transformation where the centre of the image moves slower than those in the image's boundaries [106]. Therefore, the author has used the bilinear model which is more robust to the non-linear transformation.

2.3.2.3 Graph Cut

In early vision, researchers attempt to assign labels based on different expressions such as intensity, disparity, segmentation regions etc. and to assign labels to the pixels based on noisy measurements. This labelling will be based upon the calculated energy. The route of finding the best labelling is seen as an optimisation problem. Graph Cut is a method that is used to minimise the energy and it is one of the most used techniques for visual tracking scenarios where the targetted problems can naturally be expressed in terms of energy minimisation [28], [61], [81], [86], [89] and [119]. In the last few years, the minimum cut/maximum network flow algorithms have emerged to introduce an elegant and useful method to minimise energy. We will mainly be considering how this technique was used in the tracking techniques. However, the energy minimisation was also used to solve the stereo, motion or image restoration issues in the image processing circumstances. Markov Random Fields [28] is one of the generative models that is widely used in solving the labelling problems. For each variable to get assigned with a label, it would then need to pass through the Markov property which declares that the state of each variable highly depends on the state of its neighbours [28]. After assigning the labels, a model such as the Potts Interaction Energy Model and the Linear Interation Energy Model were widely used to determine the energy's function [28]. The graph's edges are a major part in this process where they make the contours and obviously, if we have a graph with more contours (or edges), this means the graph may include more detail. However, before we go more into the min-cut/max-flow details, we would first need to know how the flow networks in the context of energy minimisation operates. Figure 2.6 demonstrates a graph with few nodes (or pixels) and shows how the nodes are connected to the neighbouring nodes and to the top and bottom terminals (terminals are special nodes which are called "source" and "sink" in the energy minimisation method(s)).

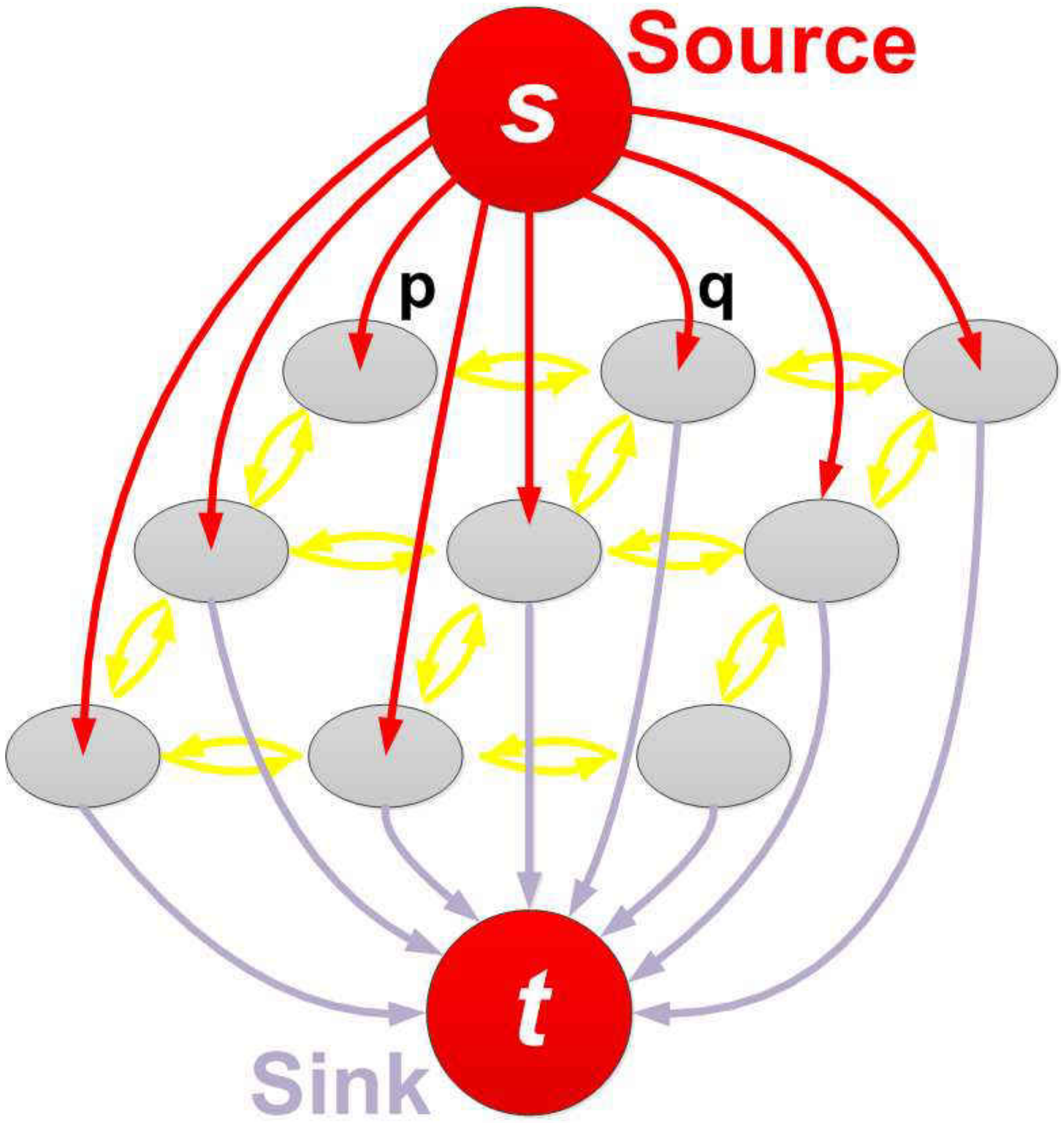


Figure 2.6 – Nodes connecting source to sink

The graph in Figure 2.6 shows two types of edges. The horizontal ones connect the nodes to their neighbour(s) (n-links) and the other type of edges connect each node to the source and sink of the graph (t-links). Horizontal (or n-links) edges correspond to the penalty of discontinuity between the pixels and the vertical edges (or t-links) refer to the penalty for assigning the label to the pixels. The fundemental network flow problem is the minimum cost flow problem, which means that we would need to define the route with the maximum flow with a minimum cost from a specified source to the node [28]. The minimum cut of a flow network is a cut whose capacity will be found to be the lowest amongst all the cuts of the network. In the Graph Cut method, the segmentation process can be the ultimate process and this is achieved by grouping the image's pixels into logical groups. In [28] the author has used the Pott Energy Function to group the pixels, which would result in the segmentation process. Work in [89] also demonstrates a technique for how to segment the different parts of an image using the graph cut methodology. In [119] the author has used the Graph Cut based active contours (GCBAC), as the advantage of using the GCBAC is that it does not require any prior global shape model. Also, unlike other active contours, it does not get stuck in local minima [119] therefore, it is not sensitive to the initial conditions. It is worth mentioning that the GCBAC was first proposed in [89]. In [119], it describes the way the GCBAC method works, where it starts by widening the current boundary into an area of interest with an inner and outer boundary. It will then start to represent the data within the two boundaries as a connected graph and divide the nodes into two groups, where the nodes on the inner boundary will be delared as a single source and the nodes on the outer boundary will be declared as a single sink. Figure 2.7 was taken from [119] and it visually demonstrates how the process is achieved where it computes the s-t minimum cut to identify the new boundary, which is clearly shown by the most right side image in Figure 2.7, resulting in the seperation of the inner from the outer boundary.

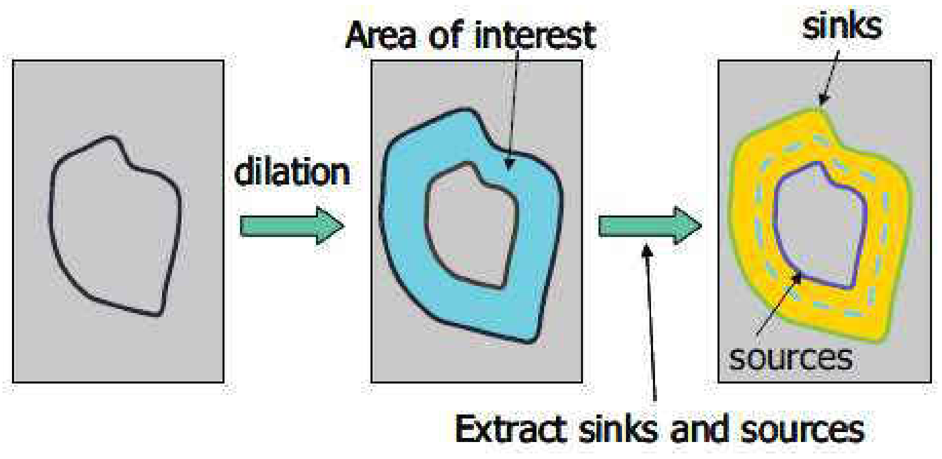


Figure 2.7 – Extract sinks and sources (s-t minimum cut) [119]

This method is useful when it is applied to an image with an object on a simple background and the initial contour is much larger than the object's contour [119].

Different papers have proposed different information retrieval to track the object's contour. For example, in [119] the author has proposed to incorporate both the image's intensity and the difference between the current frame and the previous one to track the object's contour. The way the object's contour is tracked is by applying the GCBAC on the difference data and then by applying the GCBAC again to the current frame to get the final result [119].

Tracking with the Graph Cut technique was used for many purposes. However, as the Graph Cut technique is a robust method used for global image segmentation, papers like [61] proposed the use of this technique to track multiple objects within the area of interest. The standard Graph Cut will capture all the objects within the area of interest; however, the way the post processing and filtering algorithms [92] are followed will detect the specified objects amongst other available objects within the same area of interest. With most of these applications there are some prerequisites which the user(s) are required to have before tracking takes place. For example, in [61] a user is required to mark the object that needs to be tracked in the first frame for the initialisation purpose.

2.3.2.4 Background Segmentation

Identifing the moving object(s) by applying the background segmentation is a technique that is used in many available research studies such as [13], [22], [39], [65], [107], [113], [114] and [123]. Being robust against the illumination, avoiding a non-stationary background such as swinging leaves, rain, snow as being part of the moving objects and finally the background model being able to react quickly to the changes in its background (i.e. car moved from the parking space) and update itself are all big challenges for many applications which are involved in the background segmentation process [65]. The colour similarities between the background and the foreground objects and also the increases in the foreground object's size are further challenges which are discussed in [107]. The process is mainly followed by introducing those pixels which highly differ from the background model's pixels as foreground pixels which are then to be dealt with as moving objects. Segementation and defining the moving objects' techniques vary from being as simple as frames differencing, adaptive median filtering or a more sophisticated approach such as probabilistic modeling techniques [65]. However, as the author of [65] also argues, most of the existing background subtraction algorithms in the available literature follow the steps shown by the diagram in Figure 2.8 from [65].

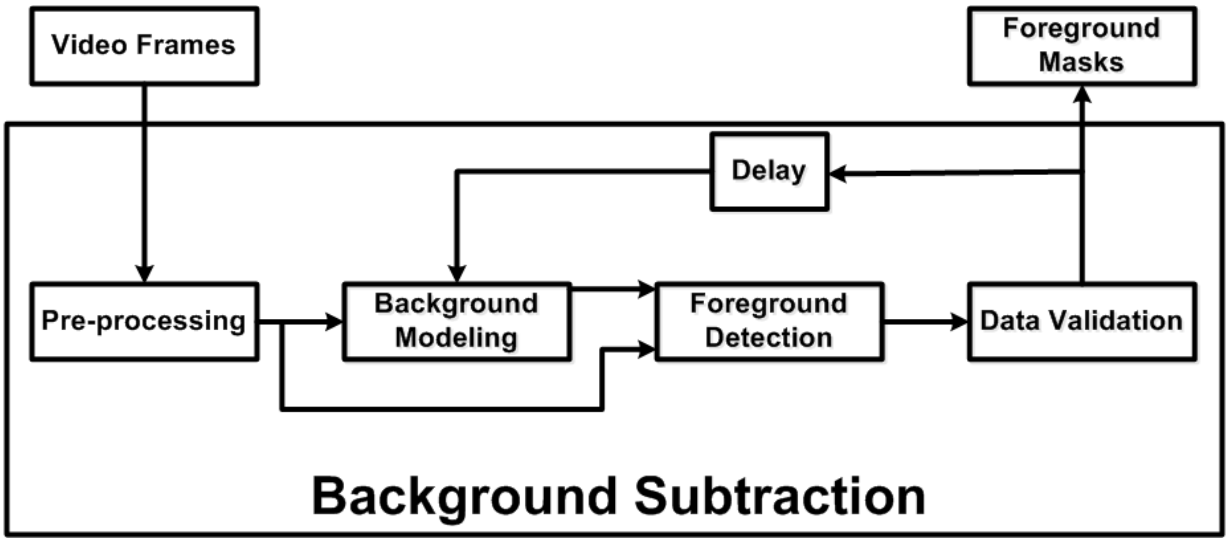


Figure 2.8 – Background subtraction steps

Following the steps from the diagram shown in Figure 2.8, the pre-processing step is used to reduce and manage the frame’s rate and size to reduce the data processing rate. As is also shown in the above diagram, the background modeling is the heart of any background segmentation process. The authors in [65] have classified the background modeling process into recursive and non-recursive categories which are each divided into different sorts of techniques. For non-recursive, the author has introduced Frame Differencing, Median Filter, Linear Predictive Filter and Non-Parametric model as four different background modeling non-recursive techniques. In addition, the Approximate Median Filter, Kalman Filter and Mixture of Gaussians were introduced as three different recursive background modeling techniques. As the author has argued, the non-recursive uses a sliding windows approach for background estimation which means it stores a number of previous frames and analyses the background features based on temporal variations of the available pixels. Median Filtering is one of the most commonly used techniques for the background segmentation process where the background estimate is defined to be the median at each pixel location of the stored frames in the buffer [65]. The Kalman Filter and Mixture of Gaussians (MoG) techniques are widely used within the recursive background segmentation's literatures but these two techniques differ slightly, as the Kalman Filter tracks the development of a single Gaussian, whereas the MoG is used to track multiple Gaussians at once. Using any of the recursive or non-recursive techniques for the background segmentation process supplies us with an outcome which should enable us to compare it against the input frame and define the Foreground pixels. After the Foregound detection, the Data Validation process is put in place, where it starts to analyse and improve the found foregrounds based on the information gained from outside the background model. Three limitations are faced in Data Validation; it does not ignore the corerelation between neighbouring pixels, the rate of adaption may not cope with the foreground objects' motion speed and ultimately, detecting the non-stationary pixels and removing them from being catagorised as a moving object is also a challenging process [65]. In [114], the author has introduced a methodology for multiple object tracking using both background subtraction and connected component analysis, where it applies the connected component analysis after the foreground mask to determine the continuous regions of pixels or blobs which results in extracting features from the found blobs. The same paper introduces a technique called Lazy Background Subtraction and Connected Components Analysis (LBSCCA) which performs this segmentation by analysing the connected pixels related to the foreground blobs. In [114] the author has used the state prediction process to determine the position of the blobs in the next frames. The Joint Probability Data Association (JPDA), together with the Kalman Filter, achieve this purpose. The JPDA was used to predict and direct the attention to the area of interest and the Kalman Filter was used to define the region of interest, which will then lead to computational cost reduction. Kalman Filtering uses the current state based on the previous blobs' states and then uses the current state to feed the prediction of the next states. In [114], besides the segmentation and the foreground detection process, the author has used the instance based regression algorithm [26] to determine the contact point between the blob features and that is to distinguish between the real body and its silhouette. With the scenario in [107], the author implemented some additional sophisticated processes, where they used the Spatial Color Gaussian Mixture Models (SCGMM) to model the foreground and background models. The author has built those models into the Markov Random Field (MRF) energy function to get it minimized and segmented using the Graph Cut technique. The advantage of SCGMM over many existing approaches is that the proposed SCGMM focuses on tracking both the foreground and background models, rather than only the foreground objects, as the foreground and the background models will compete to grap the related pixels within the frame. Also, unlike the ordinary EM algorithms where the algorithm looks at the models for the updating process, in [107], the author has used the EM algorithm to update the spatial parameters of the SCGMM models which would also result in speeding up the entire performance.

2.3.2.5 Template Matching

The statistical models [112] of specific feature(s) (i.e. skin) are widely used to make use of their colour and shape as an appearance based template for the subsequent tracking process [57], [75] and [120]. The overall template matching algorithm starts by template initialisation then determination of the searching region and ultimately, template matching and the update process [75]. In template matching, the background subtraction is used as part of the initialisation process. In [75], the author has used the template matching technique to detect and track human faces where it starts to use the skin colour and the maximum likelihood estimation (MLE) to detect the faces and use them as an appearance based template for subsequent tracking. The template matching is processed by applying the sum of squared difference (SSD) [49] to measure the difference between the target and the reference templates. However, the colour information can detect faces and some other body parts such as the hands. Therefore, in Template Matching techniques, most authors (i.e. [57] and [75]) also use shape filtering to distinguish between the similar parts. In [57], the author has relied on the contour edge detection to retrieve the shape model. Nevertheless, in most of the Template Matching techniques, the initialisation process would require a proper background definition and that is achieved by projecting the camera onto the environmental scene with no human bodies or any constant motion which would then enable the system to easily distinguish between the background and the human models.

The scenario in [120] is a bit different where the author used trivial templates to find targets in a new frame. The sparsity achieved using the $\mathcal{L}_1$ regularized least squares problem and in order to improve the robustness of the sparse representation, the author has introduced non-negativity constraints.

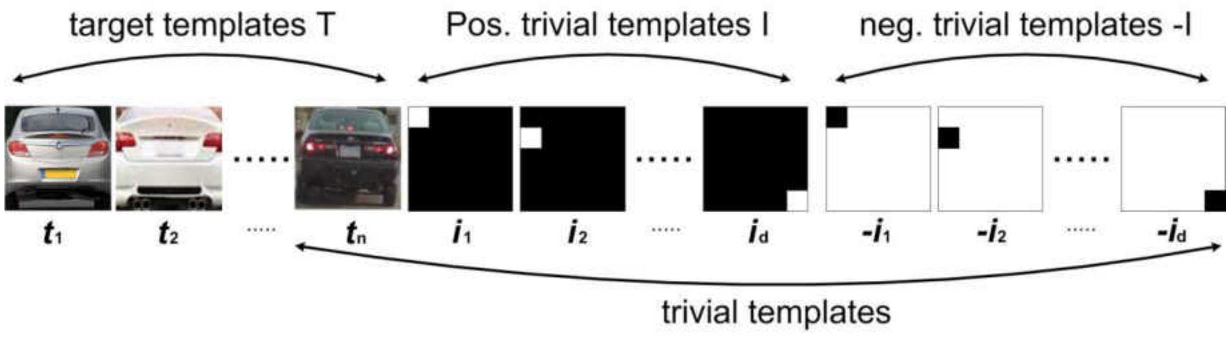


Figure 2.9 - Positive and negative trivial templates

Neverthless, as Figure 2.9 demonstrates, those constraints are implemented, as in both positive and negative trivial templates the aim is to eliminate those clutters that are similar to the target templates. This process will increase the robustness of both detection and target tracking. However, the author has used the particle filter technique to estimate the posterior state.

2.3.3 Contours

Contour based tracking is one of the fastest and most robust used techniques in detecting the moving object(s) [4], [83] and [125]. Motion-based and Segmentation-based energy are two types of useable counter based tracking techniques which are widely used in some applied research studies [83] and [125]. Snakes [83] is a widely used contour based method in research works. The Snakes method is an energy minimisation method which is highly influenced by both external and internal constraints.

Internal constraints are the image's properties that pull the Snakes toward the features (i.e. Edges, Lines). Snakes are used to resolve many visual problems including edge detection, lines, motion tracking and stereo matching [83]. The external constraints are the user interactions which force the Snakes to be near the feature of interest. In [83], the author has proposed to address finding the salient image contours (i.e. edges, lines) and to track them in subsequent frames. However, during the tracking process, the high level influences (i.e. user interaction) can still be used to push the contour model towards the appropriate local minimum. Therefore, Snakes does not resolve the entire problem of finding the salient image's contour, but they highly depend on the high level influences such as user interactions. Also, if we require a good outcome, we should place the Snakes close to the required target because the Snakes deform themselves to be consistent with the nearest surrounding contour. Nevertheless, the high level of interaction can be more intelligently designed to be automatic attention mechanisms or high level interpretations [83]. The authors in [83] have proposed Line Functional, Edge Functional and Scale space to be three different energy functions to attract the Snake. Line Function has used the image's intensity where the Edge Function used the features' edge properties and finally, the Scale space monitors the whole Snake, where if parts of the Snake get mislead then the neighbouring parts will pull that part towards the possible continuation parts of the feature. Most of the contour based designed tracking methods are fast and robust, as they mainly target the feature's edges rather than the whole region [125]. In [125], authors have proposed a method which defines the contours by detecting the edges based on the optical flow. Authors have used the Canny Edge Detector [24] to extract the required features. The way the contour extractions are processed in

[125] can differ slightly from the way they are followed in [83]. In [83] we had the internal and external powers influences detect and track the contours using the image's intensities. The authors of [125] have proposed a few steps to extract the contour and they start by restoring the lines and subtract the background then cluster those lines by the nearest neighbouring in respect to their distance and velocity. Ultimately, the contours of the clustered lines are extracted using the Snakes method. The way those features are then tracked is by defining the similarities between an object in a previous frame with the one in the current frame using the estimated positions of lines by optical flow [125]. The [27], [83], [95], [110] and [125] papers are good examples which demonstrate how the contour based or the Snake model were used for the object's detection and how they are tracked in subsequent frames. As explained earlier, the original Snake model does not get attracted to the feature if it is not close enough to the feature's edges. However, in [27], the author used the Balloons technique that enables the model to act like a balloon which is inflated by the additional forces. It passes over the edges and is stopped when it encounters a strong edge. In [125], the author has demonstrated a good addition to the Snake option. The author has designed a methodology to track the objects' state as being as occluded, reappeared, merged or separated. The author argues that if the object (or contour) disappears without reaching the frame's border, it will then be considered as occluded and if the contour suddenly appears inside the frame which matches with one of the currently available contours, it will then be counted as reappeared. However, the author has also argued that if the detected object inside the frame is found to have the lines of two different objects, then it will be counted as merged. But, if two or more objects are found to have similar lines it will then be considered as separated. More research was applied on using the object(s) contours besides the use of Level Set, such as [4], [6], [25], [38], [76] and [99]. In Level Set, the object's contour is defined by the intersection directional curves referring to each region [4]. Figure 2.10 from [4] demonstrates the

object's band with light grey in Figure 2.10.a and background band with dark grey around the object's contour that is demonstrated by a white ellipse. Sub-regions around the object's contour are represented by rectangles (see Figure 2.10.b).

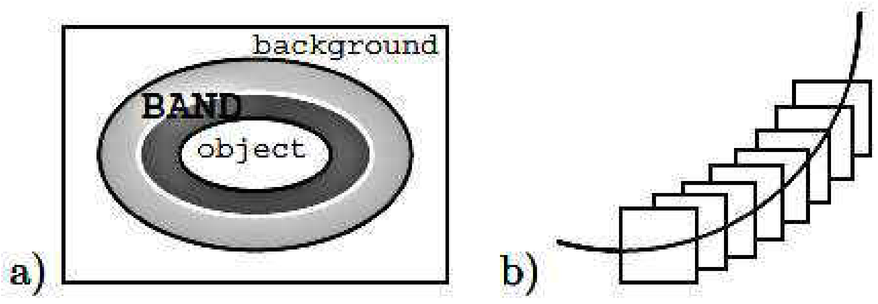


Figure 2.10 – Representing the sub-regions around the contours using the level set procedure [4]

The author of [4] has used the strategy of using a band around the boundary rather than the entire region. The advantage of using a band is a reduction in the searching space in the contour which results in a speedup of the performance and reduces the noise and possible holes in the object(s). Other advantages are the generalisation of the boundary and region based energy function into one framework and ultimately, it can adapt more to the local changes around the object and allows better reliability of the object tracking when a mobile camera is used.

2.3.4 Features

Several literatures [94], [97], [102] and [122] have used feature extraction methods for tracking purposes. The content of the "Features" section is subcategoriesd as Saliency Tracking, SIFT and KLT (Kanade–Lucas–Tomasi). However, each subcategory refers to one or more tools or methods used for tracking intention.

2.3.4.1 Saliency Tracking

Salient features are those features which are mostly noteable within the feature space, amongst others. For example, the salient feature of a flying bird is its wings and the salient features of the sky are the clouds. Salient points are also called the interest points which are obvious to the human eye and can be extracted initially to be tracked in subsequent frames [94]. Traditional approaches used the edges and corners to detect the most salient point {i.e. [97] and [102]}. Tracking those salient features will decrease the time consumption by a good factor as we save up a huge amount by processing the static views on every iteration. The authors of [44] and [94] have designed their tracking system based on the use of the salient regions. In [44], the author designed a tracking approach from a mobile platform where it learns the trackable features from the first captured frame based on the most salient points and then computes a feature vector which describes those features. The detected features from the first frame will be supervised using the attention system VOCUS (Visual Object detection with a CompUtational attention System) which was designed based on the human's visual system and detects the most salient features [44], unlike many bottom-up searches where no pre-knowledge information is required. But, in VOCUS, the top-down search is followed where the pre-knowledge of the tracking objects is mandatory. As the top-down search requires some pre-knowledge about the objects, in [44] the user selects the object by manually drawing a rectangular box around it. For the feature matching, it is not necessary to do feature matching between the regions of the reference and the region of the current frames since the top-down search (by VOCUS) will guarantee the similarity. The author of [44] has compared the Most Salient Tracker (MSR) against the Camshift [1] algorithm which demonstrated a better performance of MSR. Camshift achieved an average performance of 45% and the MSR achieved 88%, indicating clearly how much better the MSR performed. In [94], the author has focused on views with large traffic intersections, which would be expected to achieve a high volume of salient points. The author has used the Lowe keypoints and Scale-Saliency algorithms to detect the salient points, which in this case are mostly the vehicles. The output of these two algorithms are a set of features which correspond to the most salient features found on the scene. Those features will be tracked to observe the moving objects

[94]. The author has then used the graph matching approach to determine the similarities between the set of features.

2.3.4.2 Scale-invariant feature transform (SIFT)

Scale-invariant feature transform (SIFT) was published by David Lowe in 1999 [78]. The method was designed to detect and describe the local features from the captured training images. SIFT was implemented to extract the highly distinctive features from the image. After capturing a few frames to be used as training images, the next step is to build a scale-space pyramid where the captured frames are filtered and down sampled. The features will be stored in a local database where the operator will compare the found features in the new image to the stored database based on its location, position and scale and its appearance based on the Euclidean Distance metric. In literatures, the use of SIFT will guarantee to remove the difficulties of extra computation for possible illumination changes. The feature's descriptions are the absorbing details which can then be used later to search for and locate the objects in subsequent frames. SIFT would be interested in the static features which have relative positioning in between. If any of the selected features changes their positions, they would be declared as an error. But, SIFT usually selects a huge number of features within the image and if only a few of these become an error, it would not have an effect on the ultimate outcome. In [12], [69] and [77], the authors have used their tracking methods based on the use of the SIFT technique. The goal is to recognize, determine the spatial state and the relationship between the objects' positions to enable the systems operates the same way the human visual system works. Therefore, in [77], the author designed a tracking system based on the IVSEE system design which tries to initiate the early functionalities of the human visual system. However, as the author of [77] also argues, the systems which are based on SIFT can operate faster than those which are based on the segmentation techniques, but the systems with segmentation techniques provide much more precise information about the extent of the objects [77]. As in [69], the tracking procedure is achieved by letting the user select the target and both the SIFT and the Kalman Filter use the chosen target to perform the tracking process. When the user selects the object(s), the SIFT features within the objects location are stored. Figure 2.11 was taken from [69] and demonstrates how the procedure of an algorithm starts by letting the user select the target. Next will be when the Kalman Filter starts its interaction process with the stored SIFT features. However, as the author of [69] also argues, in order to use the Kalman Filter and gain the best possible performance, we should expect a constant motion of the object (s) through the whole frames' sequence. This interaction allows the Kalman Filter to predict the next object's position.

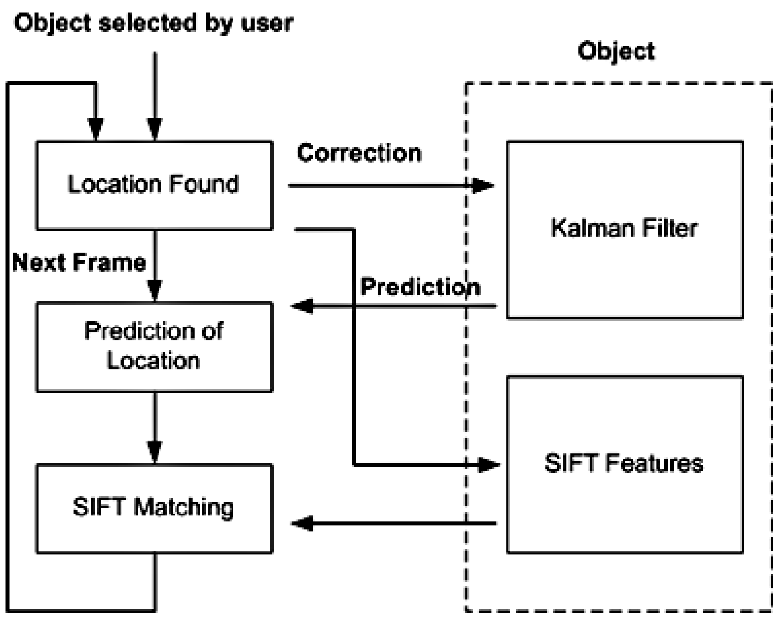


Figure 2.11. Features extraction using the SIFT technique [69]

As the SIFT keeps the Kalman Filter frequently updated, a single mistake may result in further wrong predictions in subsequent frames. More literature like [12] also used the SIFT technique in different scenarios, such as recorded video where the technique is used to determine the inter-frame motion through consecutive frames in the sequence.

2.3.4.3 Kanade–Lucas–Tomasi (KLT)

The Kanade–Lucas–Tomasi feature tracker (KLT) is a method of extracting features from the image's space. It is widely used as a differential method for estimating the optical flow and it was first developed by Bruce D. Lucas and Takeo Kanade. However, in the scientific terminology, the optical flows are the noticeable motions, surfaces and features' edges within the scene. For example, as we drive across the motorway, trees, buildings and signs are counted as optical flow. As a consequence, the KLT algorithm assumes a constant flow of pixels in the local neighbourhood which solves optical flow equations using the least squares criterion [14] by combining several details from nearby pixels. Studies [66] and [79] provide more specific KLT algorithm details where [62], [126] and [129] use the KLT algorithm for tracking purposes. In [130], the author has used the implemented version of KLT algorithm in the OpenCV platform. As the author explains, the method starts by converting the reference and the current images into black and white images then determining the useful features. The "useful" terminology here means more distinctive, for example, choosing corners is more important to tracking than inner pixels so it will start by tracking them from one frame to another. Another usage of the KLT algorithm was demonstrated by [126] where the author has designed a greenhouse sprayer navigation robot which uses the KLT to detect the features for its visual odometry. The visual odometer was designed to estimate the vehicle's position and orientation within the world's co-ordinates by selecting a few features on the ground then allocating the boxes of KLT features within a larger box to enable each KLT box to search its neighbour pixels, with the aim of determining the similarity within the subsequent frames. The author of [126] designed the KLT boxes as 7×7 pixels and the search boxes as 25×25 pixels. If the platform loses the KLT features for any possible reason then it will start to search for another useful KLT feature which can replace the lost one. However, as the author has also explained, we would only need 3 feature locations to determine the position and the possible rotation of the platform. However, the author has used 5 feature locations to increase the process accuracy. The big advantage of using these sorts of systems is related to human's health and safety procedures where people would not be exposed to dangerous chemicals. Also, the system is able to operate 24 hours a day with the same accuracy that it started with.

2.3.5 Adaptive Methods & Filters

In the previous sections, we have seen how some methods were designed to adapt themselves in order to keep the their models frequently updated [1]. In [19], [85] and [118], the authors have introduced more specialised adaptive techniques for tracking purposes. The author of [85] argues that it is most of the current tracking methods using predefined distance metrics which are likely to guarantee the exact match in all cases. If we encounter some strong features which are highly discriminative, then using distance metrics, such like Euclidean Distance, would still provide an acceptable outcome. But in cases where the features are hardly distinguishable from its background, the Euclidean Distance metric may not be the correct selection. Therefore, the selection process of appropriate distance metrics for robust visual tracking was then initiated. The author has taken the supervised and unsupervised approaches as the two main categories of general distance metrics learning. The unsupervised distance metrics are those where the relationships among the observed data are preserved. PCA [82] is a good example of unsupervised distance metric learning. The case with the supervised metrics is different as the method is fed by a pair of similar and dissimilar data for a better discrimination process. The author of [85] has used the supervised scenario of distance metric learning where both positive and negative data are provided. These two sets of data are specified when the feature of interest is specified. Then the feature vector will be extracted and labelled as positive data, whereas the regions far away from that feature will be labelled as negative data. Ultimately, for the tracking purpose, the learned metric will be used to measure the distance between the target and the candidates. This measurement process is achieved by the use of a gradient-based method rather than an exhaustive search, which would certainly decrease the processing time. Adapting the correlation methods is another used procedure for object tracking. The filter based trackers do get trained on a number of example images and correlation filters, where the target(s) are initially selected using a tracking window centred on the object in the first frame [19]. The author explains that in order to speed up the tracking process, the correlations are computed using the Fast Fourier Transform (FFT) algorithm. The Bolme [19] is an adaptive method which was designed to adapt the correlation filters. The author argues that the correlation filters are able to track complex objects through rotation, occlusions and other distractions over 20 times the rate of current state-of-the-art techniques [19]. As in [85], the filters would also need to be trained from a single frame and kept updated for any possible object's appearance changes. The author of [19] introduced a new correlation filer called Minimum Output Sum of Squared Error (MOSSE) filter. The author also argues that the trackers which are based on the MOSSE filter are robust to possible changes in lighting, scale, pose and a few other types of changes. MOSSE is an algorithm which produces ASEF-like filters [32] but with much less training images, which would make it a faster process. The author also argues that the tracker which is based on the MOSSE filter can operate up to 669 frames per second, whereas trackers like Incremental Visual Tracking [31], Robust Fragments-based Tracking (FragTracker) [100], Graph Based Discriminative Learning (GBDL) [117] and Multiple Instance Learning (MILTrack) [9], in addition to their complexities, may hardly perform up to 20-30 frames per second. To measure the correlation, a measurement of peak strength called Peak to Sidelobe Ratio (PSR) is used.

2.4 Conclusion

The content of this chapter was divided into two major divisions ("Input" and "Tools & Methods"). The "Input" section discusses various cameras and colour spaces used within the tracking systems. The second section ("Tools & Methods") was subcategorised as Probability distribution, Contours, Features, Edges & Segmentations and Adaptive Methods & Filters, though the listed tools/methods in each subcategory share the similar methodology used in their tracking systems. However, we would need to test or study the strengths and weaknesses of the presented methods to assist us with designing the ultimate tracking algorithm. For example, with probability distribution methods, the risk of non-gaussian (or non-linear) distribution is always expected. Though, methods such as Mixture of Gaussians [65] were designed to resolve the single gaussian issues, it might introduce a further computational cost. Feature tracking is another widely used low cost technique but its complexity appears with the feature extraction procedure beforehand.

Contour based techniques are also broadly used tracking method that uses the edge tracking technique to identify the object's contour. This clarifies the likeness between the contour based and the edge tracking methods. However, a major issue emerges if the fast displacement misses the possible correlation which can guide us back to the target. Though, few methods (i.e. "Snakes") were developed to enhance the contour based tracking procedure. Segmentation (i.e. background segmentation) is another technique used for tracking purposes but depending on which method to use, this technique can turn out to be expensive or undependable. We have also presented various adaptive techniques that are used within the tracking algorithms. This adaptation helps to enhance the system performance on the runtime period.

This review shows a wealth of techniques that have been developed but also shows that there is a need for a lightweight real-time algorithm for motion compensation of platforms. The research question stated in chapter 1 requires us to examine image-based stabilisation that can be selected empirically using pixel-wise comparison. In order to address this we must now examine appearance comparison techniques which the next chapter will concentrate on.

Chapter 3

Appearance Comparison

3.1 Preface

This chapter aims to illustrate the various investigations made into the appearance comparison techniques that might be useful to test the research question asked in chapter 1. The chapter will demonstrate the strengths and weaknesses of some of the most widely used appearance comparison techniques. However, one technique will be chosen for use in the real time experiments.

Figure 3.1 shows an image which visually demonstrates that both the camera and the human being are looking at the same view. By nature the human's brain is clever enough to determine the angle that the human's eyes are required to rotate in but the computer must be programmed to track using the precise details of what to look for and where. In order to achieve this, computers demand that some strategies should be followed and more cleverly designed strategies will help to achieve better tracking. The common point between the human and the computers is the fact that both are needed to store an initial view and keep comparing the new views with the one stored in the memory. On the computer's side, the initial view is the captured and stored sections which can be part of or the entire captured image. We call these captured sections "Patches", where the patch from the reference image with the one captured from every current image is to be compared to determine the distance between the two images.

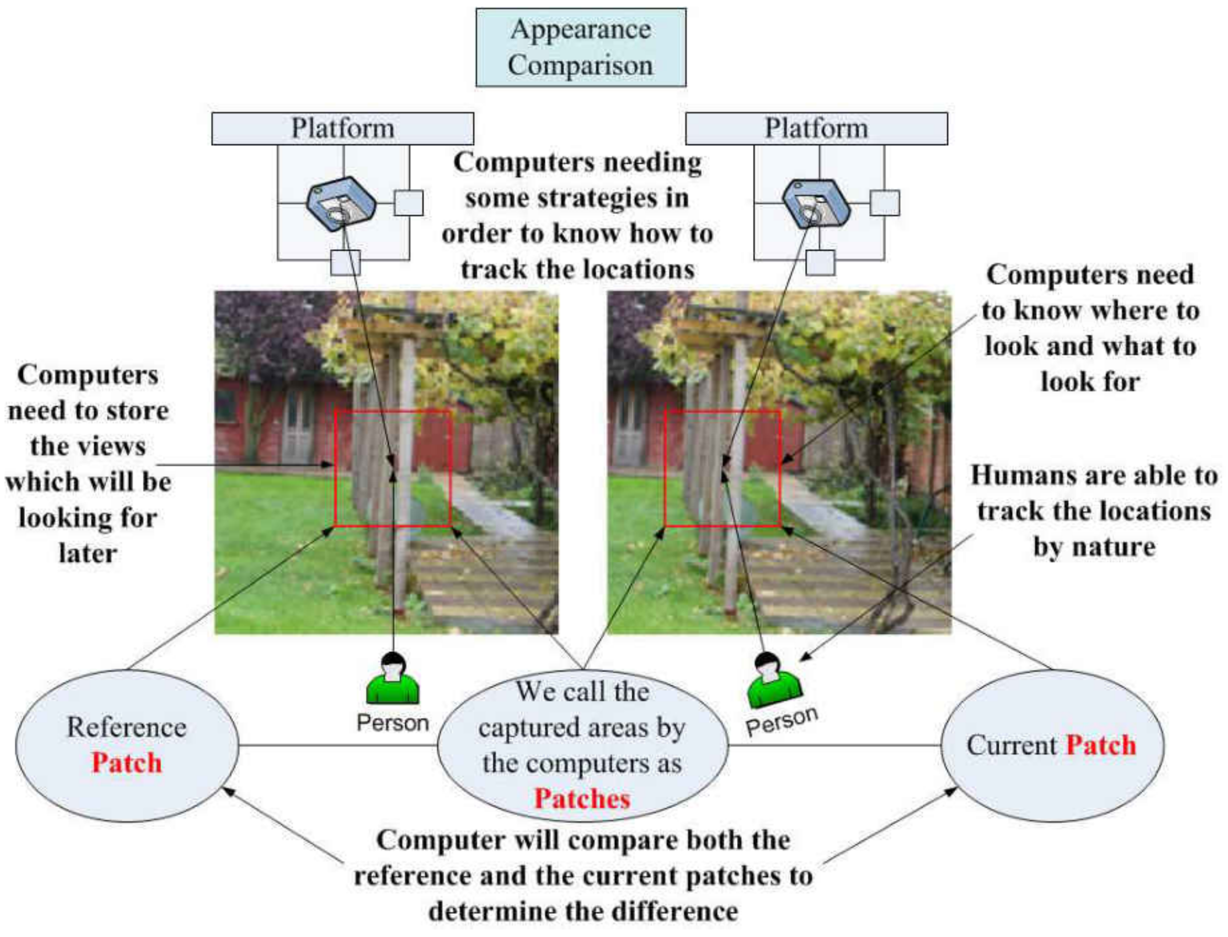

Appearance Comparison
Platform
Platform
Computers needing some strategies in order to know how to track the locations
Computers need to know where to look and what to look for
Computers need to store the views which will be looking for later
Humans are able to track the locations by nature
Person
Person
Reference Patch
We call the captured areas by the computers as Patches
Current Patch
Computer will compare both the reference and the current patches to determine the difference

Figure 3.1 – Tracking the original location

Selecting how and where to locate the patches is an essential procedure which needs to be followed carefully. We start to explain this procedure by using simple designed artificial images which will provide us with a good base to understand how the Error Surfaces will be generated over the real world images. In section 3.2, we will use various artificial or real world images to generate a range of error surfaces to help us with classifying a variety of reasons affecting how the error surfaces are shaped.

3.2 Error Surface

We begin by describing how the Error Surfaces are technically generated by duplicating similar artificial image and call one the "Reference" and the other "Current" (see Figure 3.3).

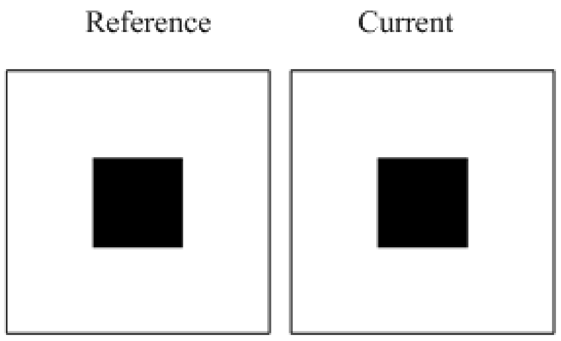


Figure 3.3 – Comparing similar images

The first step is to capture and store a patch from the most central location of the reference image. The left hand image of Figure 3.4 demonstrates how this process was carried out. However, the right hand image presents how the reference patch is located and shifted over every possible location of the current image.

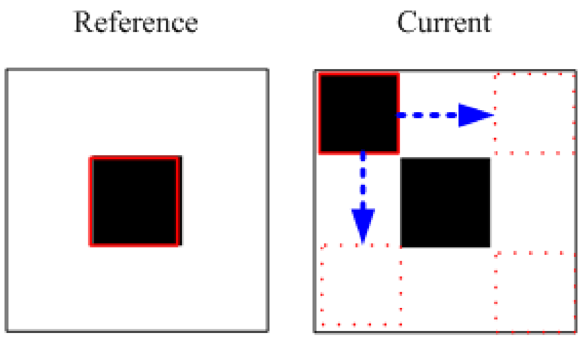


Figure 3.4 – Reference patch shifting on the current image

The comparison compares every pixel from the reference patch with every corresponding pixel on the current patch. Figure 3.5 shows two captured patches which demonstrates an example of how they can be completely matching or completely mismatched. The left hand side patches are two patches of black and white which shows a situation when the central patch from the reference image is applied over the complete disparity areas such as the top left hand corner of the current image (Figure 3.4).

Figure 3.5 – Representation of both set of patches

There are many potential strategies for the determination of the distance between pairs of images and there are few papers which directly address this [2]. For the purposes of this work, we have used the computationally low complexity Euclidean Distance to calculate the distances. All work presented here uses the RGB colour space, although we are aware that other colour spaces may have different and possible beneficial properties. Other colour spaces such as CIE L*a*b* [134] are more specialised in excluding the illumination from the original colours which may help in scenarios where the colours' properties are changed due to a change in the surrounded lighting level. For example, in a real world environment, if the lighting level increases, the red can become orange or pink which would differ from the properties of the original colours, but these issues are resolved with other colour spaces such as CIE L*a*b*. However, because we intend to apply the experiments in real world environments, the use of RGB colour space will further introduce the affect of external issues (i.e. lighting levels) to the work's performance. A calculation of the Euclidean Distance between a patch of an image of size $h \times w$ is performed using the following formula;

$$\forall i, j \rightarrow d(Ii, Ij) = \sqrt{\sum_{k=1}^{h \times w} \sum_{l=1}^{c} (Ij(k,l) - (Ii(k,l))},$$

Where h and w are the patch's height and width and k represents the current pixel. The *Ij(k, l)* and *Ii(k, l)* are the l^{th} colour component of the k^{th} pixel of images *Ij* and *Ii* respectively [11].

We simply apply this formula repeatedly to the patch and image whilst displacing the patch with respect to the image before each comparison and recording the result for each location. This process generates a distance for each tested position of the patch in the image. These distances can then be used to populate an Error Surface where the displacements in the image represent the *x* and *y* dimensions and the distance is used as the *z* dimension (height).

Figure 3.6 presents an example of the Error Surface that is generated by running the Appearance Comparison method using the Reference and the Current images in Figure 3.4. Arrows demonstrate where the patches' locations match on the Error Surface. The minimal point should represent the most possible matching between the two reference and current patches.

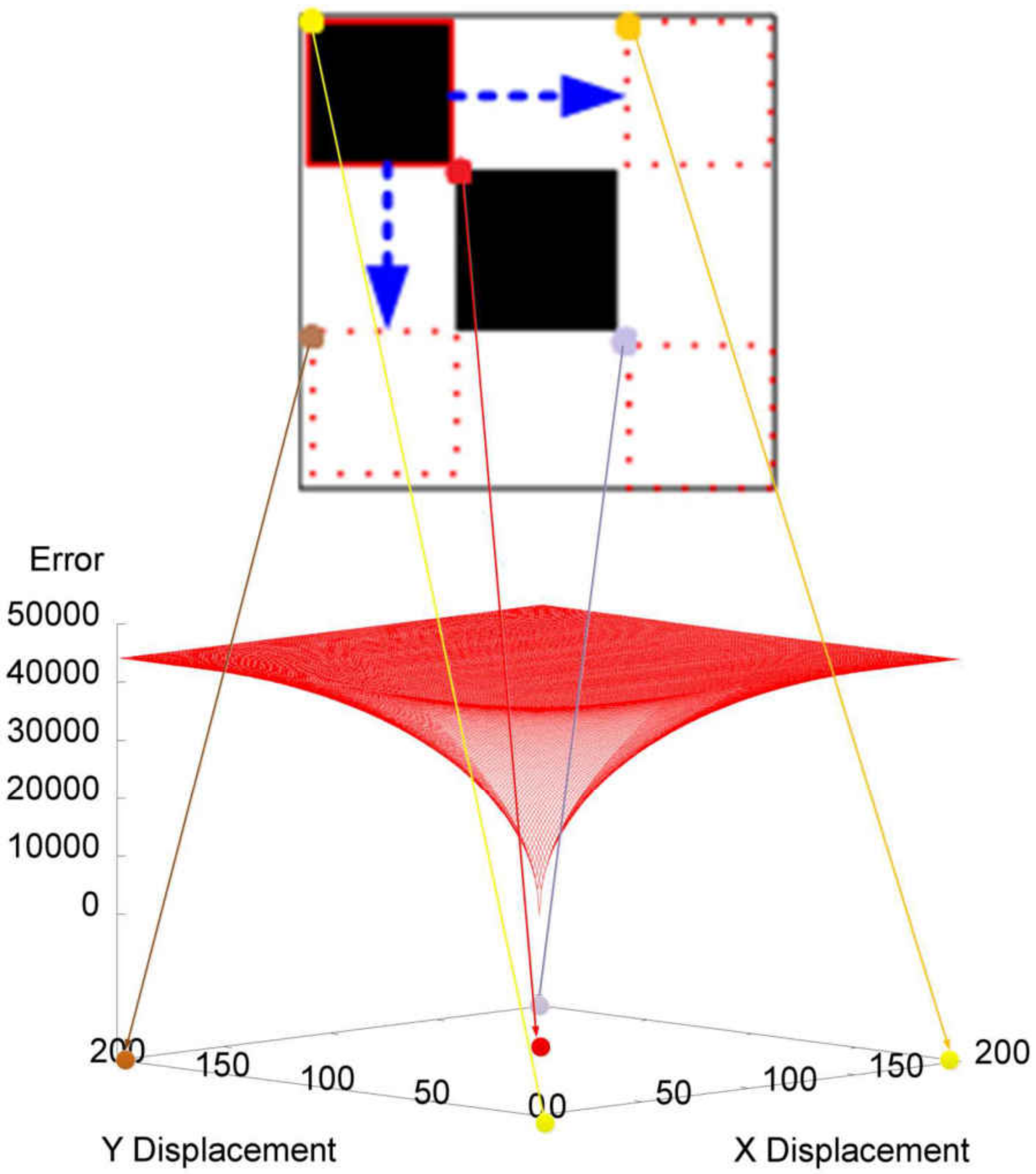


Figure 3.6 – Error surface generated as a consequence of appearance comparison process

Gathering the most information possible is the main issue that is usually considered in most applications that require some feedback from the image processing side. In our case, more details are clustered by having a larger patch and the first issue that requires our investigation is the patch's size and its initial location which would have an effect on the ultimate outcome.

3.2.1 Patch Positioning Strategies

We begin by conducting some experiments on how the patch size and location affects the shape of the Error Surface. The shape of the Error Surface is a major part as it introduces the possible navigation process to find the minimum on the surface and this is an indication of the best position for the patch to be located. We begin to use artificial and real world images to describe how the Error Surfaces are generated and how their shapes are so important for the ultimate outcome.

As Figure 3.7 shows, we start with two artificial images to present the various shapes of the error surfaces. Figure 3.7.a presents an image with a black box that is larger than the captured patch located in the central position. Figure 3.7.b presents an image with a vertical line crossing the most central position from the top to the bottom of the image. The two images are made up of black and white colours only. However, to assign pure black or white colours, we wrote a code to assign every pixel with either (0, 0, 0) or (255, 255, 255) as RGB values represent the black and white colours. For example, in Figure 3.7.a, we assign every pixel with (255, 255, 255) except the surrounding square area from (50, 50) up to (250, 250) in the X and Y position, making a square shape larger than the captured patch. On the other side, for Figure 3.7.b we assign every pixel as (255, 255, 255), except the surrounding rectangular area from (140, 0) up to (160, 299) in pixel position which will be assigned as (0, 0, 0).

In both images, the surrounding square area from (100, 100) to (200, 200) were captured and kept as reference patches. In both images, the captured reference patches suffer from a high similarity across the same image. This has caused an indistinct global minimum that leads to ambiguity in finding the optimal camera location.

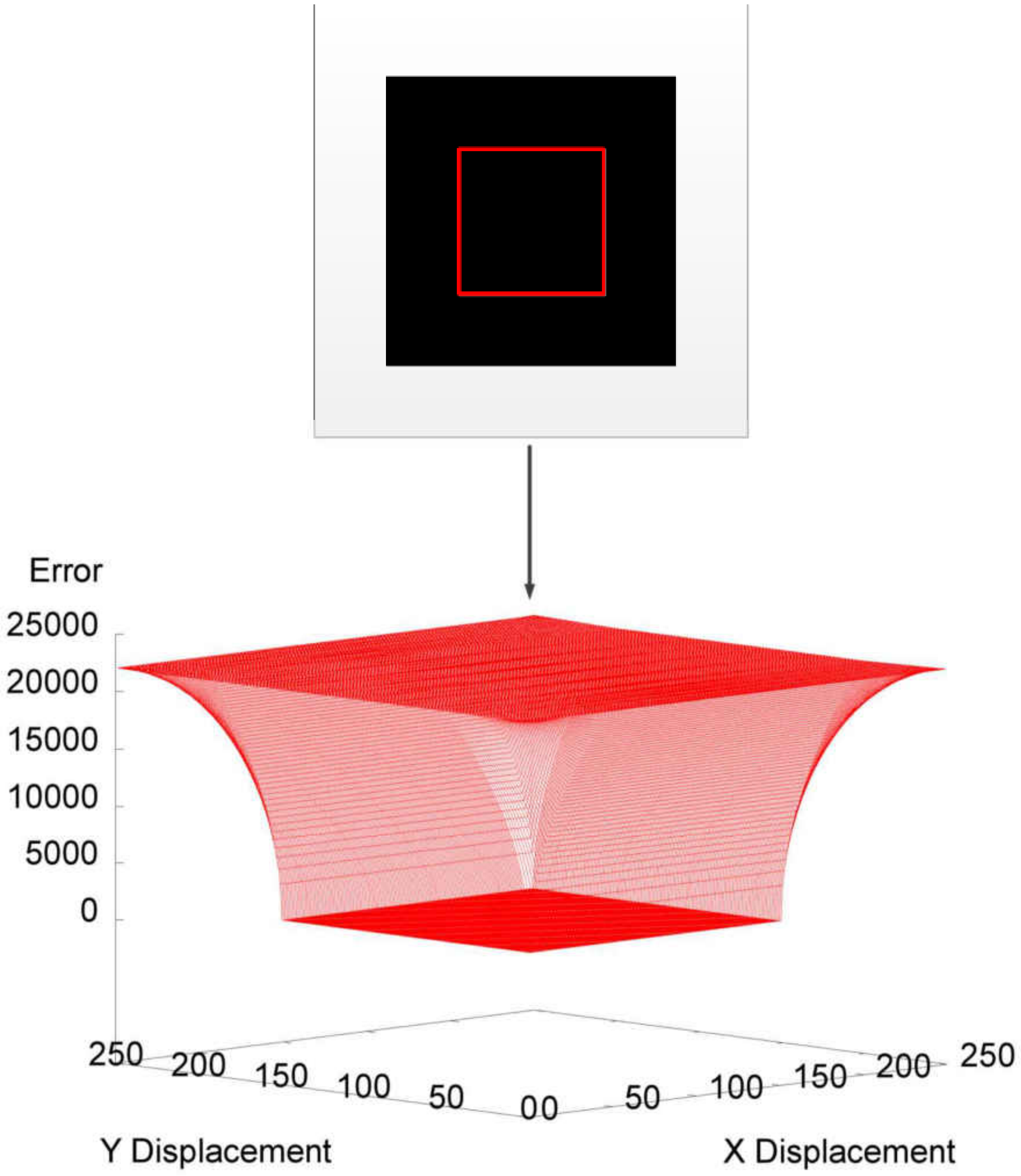


Figure 3.7.a – Error surface generated using an image with large black box in the central position

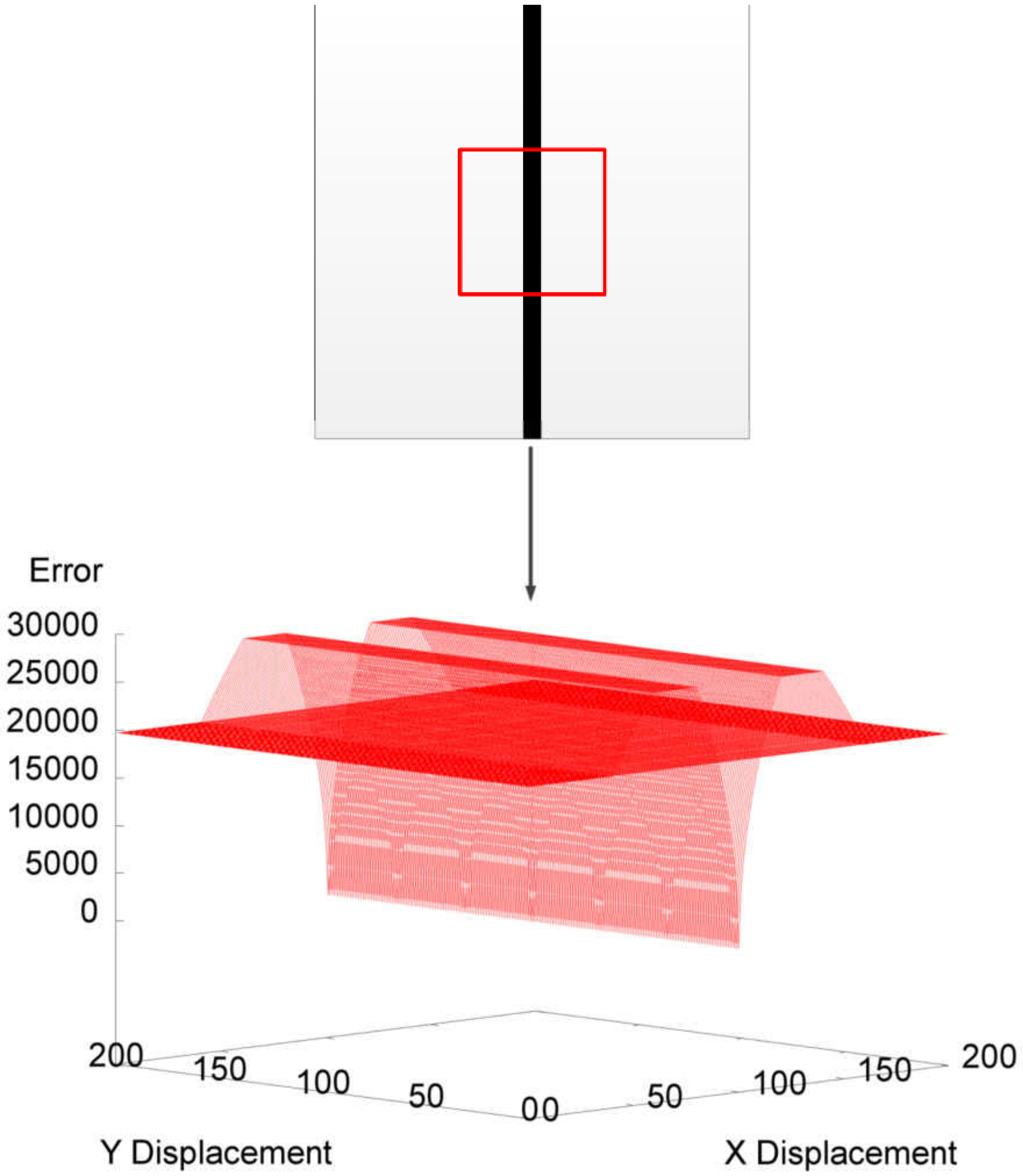


Figure 3.7.b – Error surface generated using an image with vertical black line in the central position

The above comparisons were produced based on some artificial images, but is this the same in real images? The above comparisons of artificial images clearly show that the final outcome depends on the patches we target as well as the image that we process. In real images small patch movements may cause large changes in the image's characteristics and may result in dramatic changes in error value and therefore very steep error surfaces. The challenge is therefore to find a sampling strategy that minimises this tendency whilst maximising the likelihood of generating a well-defined and unambiguous global minimum. Thus, establishing what regions to target when dealing with real world images will be the key to generating well-behaved error surfaces. In order to test these ideas we have applied these techniques to some real world images. Repeating colour patterns, large regions of constant colour and complex textures are all features which need to be assessed and thus images containing these features have been selected for testing (see Figure 3.8).

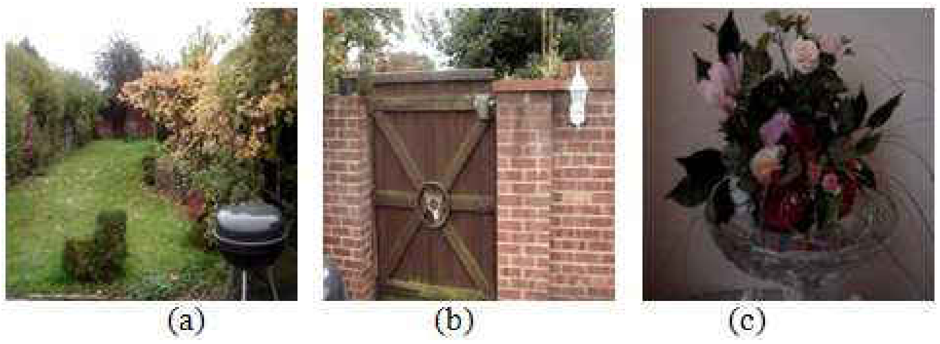


Figure 3.8 - Three different testing images: (a) Garden, (b) Door, (c) Flower

What and where to target on the image is the current focus of our work. We have tested various techniques for patch sizes and patch positioning.

3.2.1.1 *Fixed Locations*

We begin by investigating the "Fixed Locations" strategy which starts by allocating patches in certain locations. By locating patches in various locations, we will examine the possible difference that error surfaces are presented.

This strategy has been tested using three different layout patterns: *Central*, *Individuals* and *Merge*. Each was used to identify the patches' locations. Each technique has different numbers of patches, patch sizes and patch locations. The patches' properties will help us categorise the effect on the shape of the error surfaces.

Applying different techniques to the same image will result in different error surfaces because different strategies capture different views and perform the calculation process based on different pixel selections.

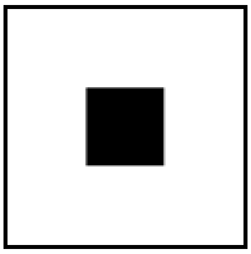


Figure 3.9.a – Patch's location in the central strategy

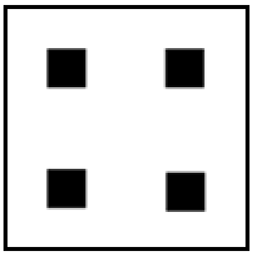


Figure 3.9.b – Patches' location in the individuals' strategy

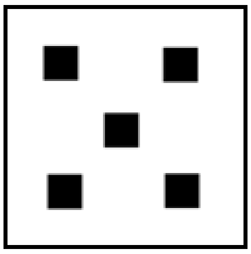


Figure 3.9.c - Patches' location in the merge strategy

Figures 33.a, 33.b and 33.c visually show where each strategy is targeting and what pixel samples it includes when applying them to images. For the appearance comparison process, patch(s) will be equality shifted to the most top left hand position iterating throughout the image's pixels down to the most bottom right hand position. However, every strategy has a different range of freedoms to iterate throughout the image, for example, in the central strategy, the captured patch is *100* × *100* in the most central position of the *300* × *300* image and therefore the patch can be shifted by

(-100, -100) up to (+100, +100). Unlike the central strategy, the individual strategy contains four *50* × *50* patches where the top left hand patch is located on position (50, 50) and the bottom right hand patch is located on (200, 200). This means to iterate all four patches jointly throughout the possible pixel positions, we will need to shift all patches by (-50, -50) up to (+50, +50). The merge strategy is quite similar to how individuals are designed and performed, except that we have five patches of *40* × *40* where the top left hand patch is located on position (40, 40) and the bottom right hand patch is on (220, 220), which means to iterate all four patches jointly, we will shift all five patches by (-40, -40) up to (+40, +40).

Examining Figure 3.9.a, the central strategy shows when applying the central strategy to the garden image (Figure 3.8.a) that the grass will occupy most of the patch. By looking back at the artificial images we can see that large single colour regions such as the big black square can lead to a lot of similarity and can have many close minimal points. The grass is thus a poor region to target as the garden image includes a lot of grass and is likely to yield a wide ambiguous global minimum. The situation is somewhat similar when applying the central strategy to the door image but rather better when applied to the flower image. The central patch in the door image also contains little variation in colour and there is the possibility that the error surface will be very flat in this region. This strategy worked best on the flower image, primarily because of the variety of colours captured by the central strategy in the flower image. In Figure 3.9.b, the individuals' strategy, the central box is divided into four equal sized parts and distributed around the diagonals. In the garden image this distribution is beneficial because it leads to collecting more variation and results in a smoother error surface with a well-defined minimum. Edges are important in terms of these sorts of variation and by examining the regions of the images captured by the individuals' boxes (Figure 3.9.b), we can see that for these images we tend to gather more varieties of colour than with the central strategy. This variety helps with producing error surfaces which descend more reliably towards the global minimum.

Figure 3.9.c shows that the situation with the merge strategy is quite similar to the *individuals'* strategy except that we decreased the four patches size and included an extra patch in the central position. The extra patch we added in the centre still only contains grass for the Garden case, but the other patches sample other colours and offset this effect.

Figures 3.10, 3.11 and 3.12 show a comparison of some representative regions of the error surfaces for all three strategies for all three images. Each figure contains three overlapping surfaces representing the *Central, Individuals* and *Merge* strategies.

Figure 3.10 shows the error surfaces for all three strategies when applied to the Garden image. It shows that the *Individuals* and *Merge* strategies have a larger gradient descending area than the surface which was generated using the Central strategy. They both have better surfaces than the *Central* one (which is the lowest, flattest surface) because of the larger slope they produced. This is because the *Central* patch mainly covers grass areas, which is homogeneous in colour and is widespread. The situation differs from one image to another and the shape of the error surfaces differs according to the image characteristics.

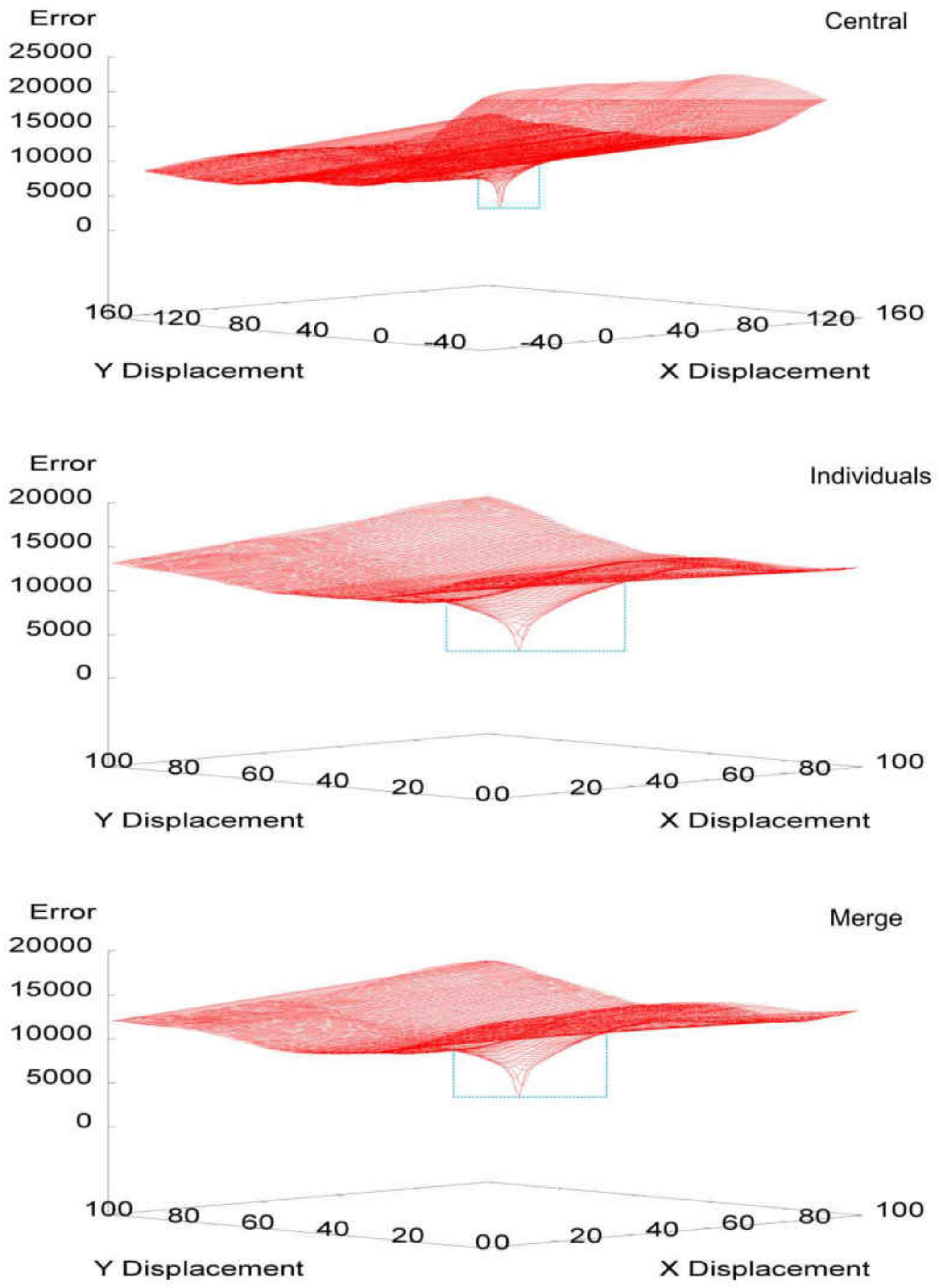


Figure 3.10 - Visual comparison for different error surfaces when applying different strategies to the Garden image (top is Central strategy, middle is Individuals, bottom is Merge). Gradient descending area is shown by dashed blue lines.

Figure 3.8.b shows an image of a door with a brick wall surrounding it. It has a repeating pattern in the brickwork and a contiguous region of colour in the door itself: two features that we expect to cause problems. Local minima are likely to be generated by the brickwork and flat regions by the contiguous colour regions. Figure 3.11 shows the error surfaces when applying all three strategies to this image. The *Central* plot is less smooth than the other two (and shows some ripples and local minima), but the *Merge* and *Individuals'* error surfaces are more smooth and have a larger descending area. The door has repeating brickwork patterns (causing the ripples) and targeting anywhere within the (contiguously coloured) door may reduce the slope. The *Merge* and the *Individuals'* patches happen to be located mostly in the corners where we have some colour variety: moving the patches in these areas generates a better error surface.

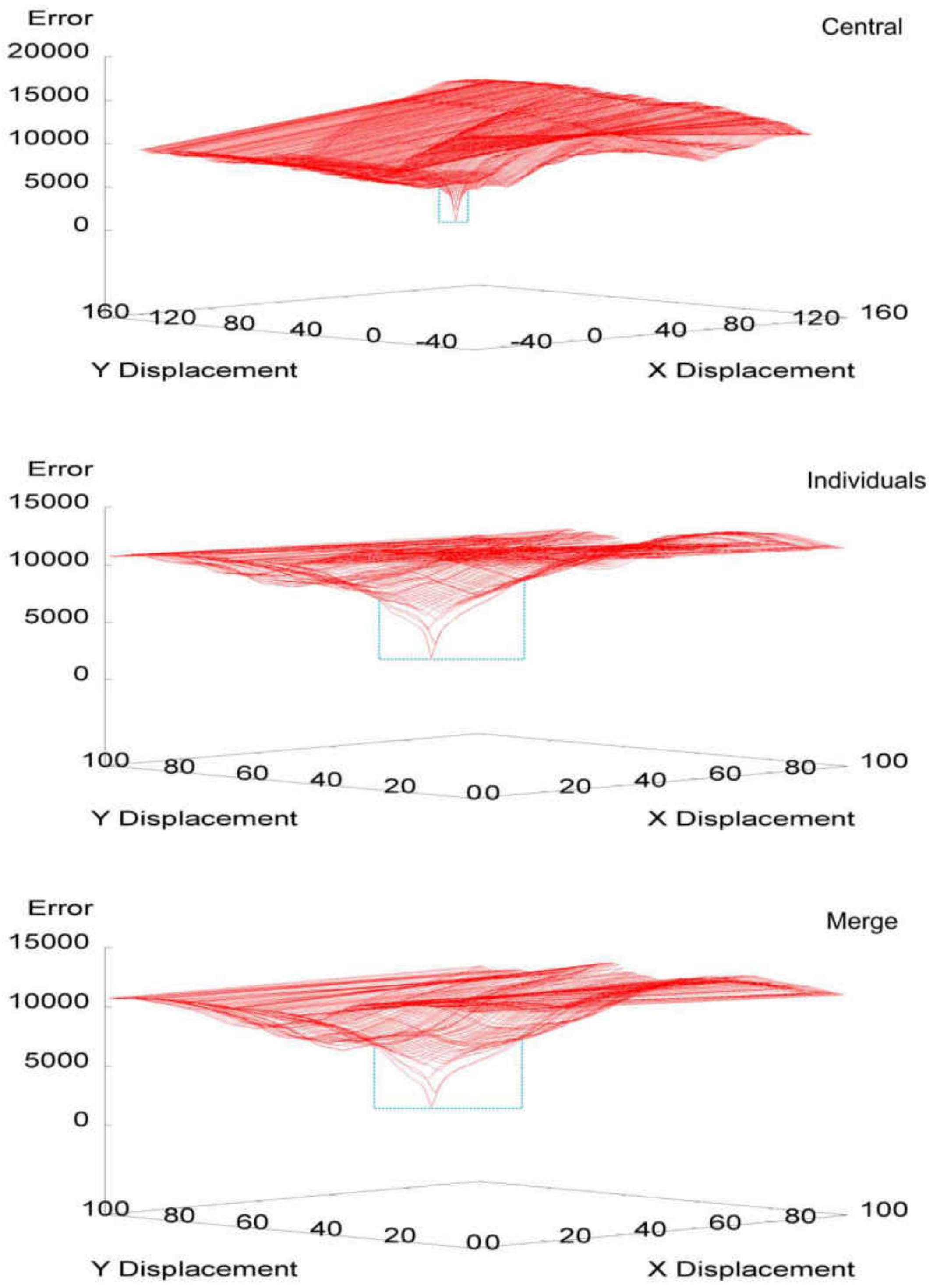


Figure 1.11 - Visual comparison for different error surfaces when applying different strategies to the Door image (top is Central strategy, middle is Individuals, bottom is Merge). Gradient descending area is shown by dashed blue lines.

Figure 3.12 shows the error surfaces when applying all three strategies to the Flower image. As in the previous two scenarios, the Individuals and Merge strategies have a larger gradient descending area than the error surface which was generated using the Central strategy.

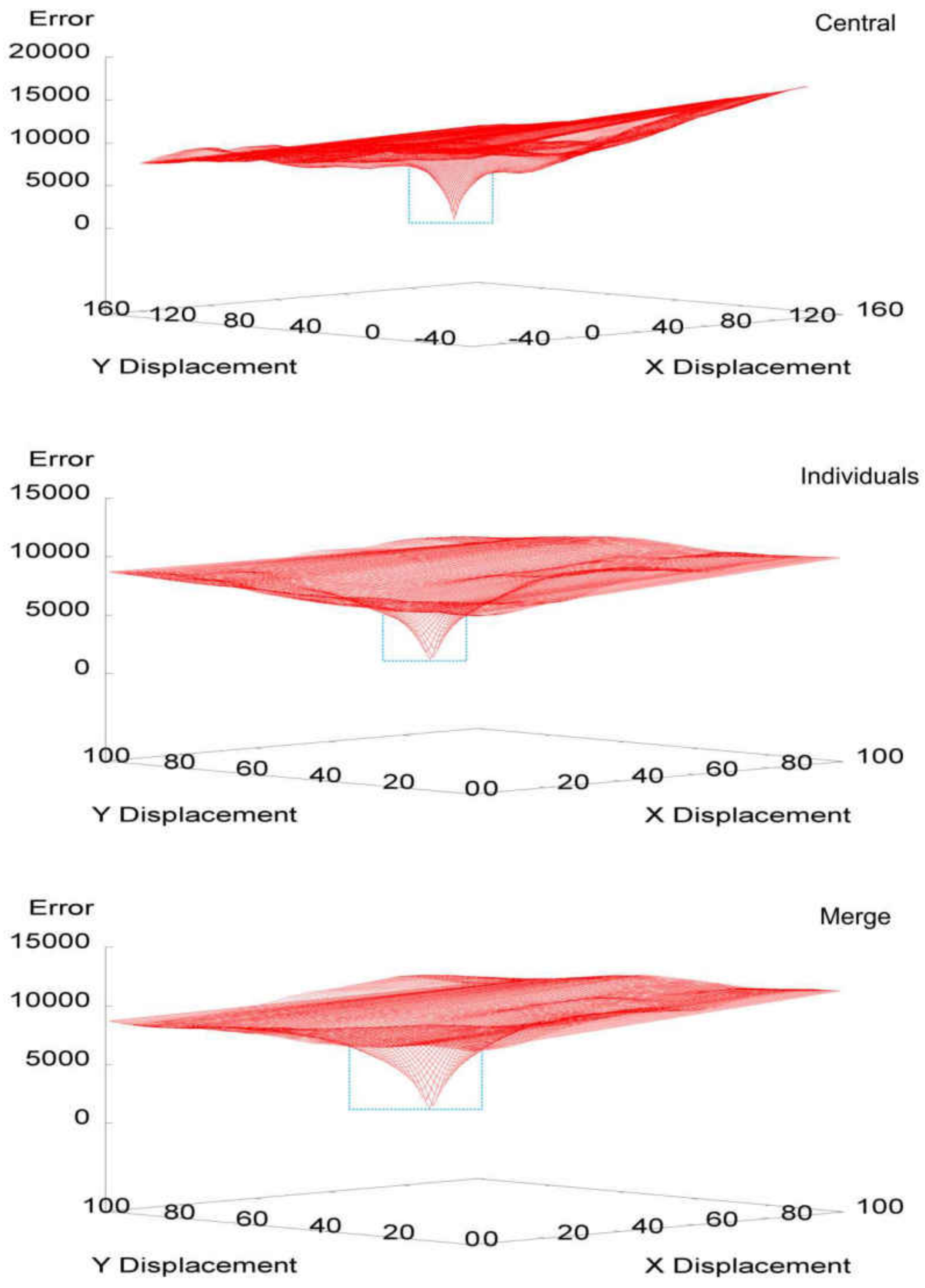


Figure 3.12 - Visual comparison for different error surfaces when applying different strategies to the Flower image (top is Central strategy, middle is Individuals, bottom is Merge). Gradient descending area is shown by dashed blue lines.

Figures 3.10, 3.11 and 3.12 have demonstrated the use of Individuals and Merge strategies as being more useful by generating error surfaces with a wider gradient decent towards the minimal point. This shows the significant reason for having various regions captured in the appearance comparison process. Region varieties will provide a better chance of getting fewer region repetitions which results in an enhanced appearance comparison outcome.

3.2.1.2 Edges

Edges (such as those in the brickwork in the door image) play an important role which can have a significant effect on the shape of the generated error surface. There are many ways of defining edges and we define them as places where there is a sudden variation in brightness because that variation might be helpful in identifying the minima on the error surface. There are numerous algorithms to identify edges and they vary significantly in terms of their performance, speed and accuracy. These techniques have been used for different purposes such as object tracking [2] and image comparison. There are also many applications to track moving objects such as vehicles which were based on these techniques [133]. Sobel [4], Moravec [6] and Robert [7] are examples of these widely used techniques which we have tested and applied to the images to see if any algorithm will help to detect patches that are suitable for generating error surfaces for minimisation and stabilisation.

For simplicity we converted the images to binary before applying the edge detection algorithm. This will identify the edges more accurately and produce less noise [8]. Figure 3.13 shows the results of the process: the grey areas in Figure 3.13.c are the edges that are detected.

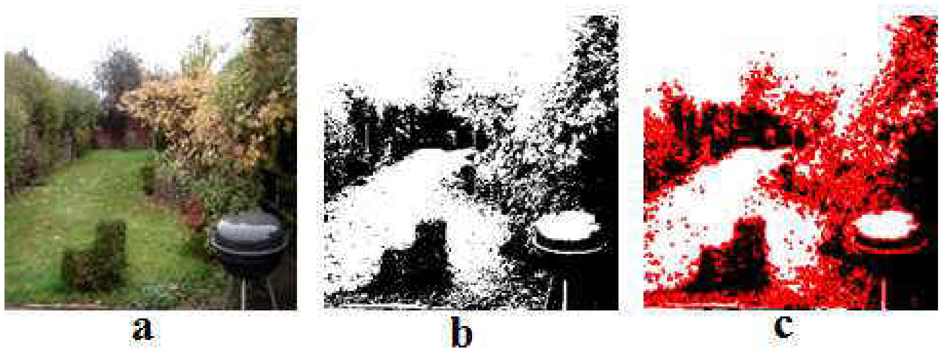
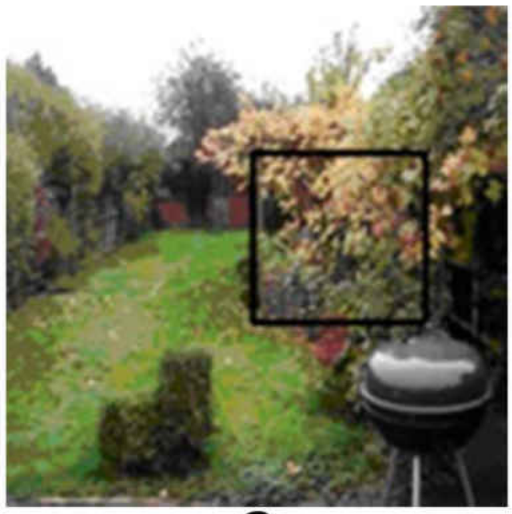


Figure 3.13 - Converting the image (a) into binary (b) then detecting the edges (c)

The next step is to identify the "best" patch and in our case we chose areas with the largest number of edges. The patch selected is shown in Figure 3.14.a and the resultant error surface in Figure 3.14.b.



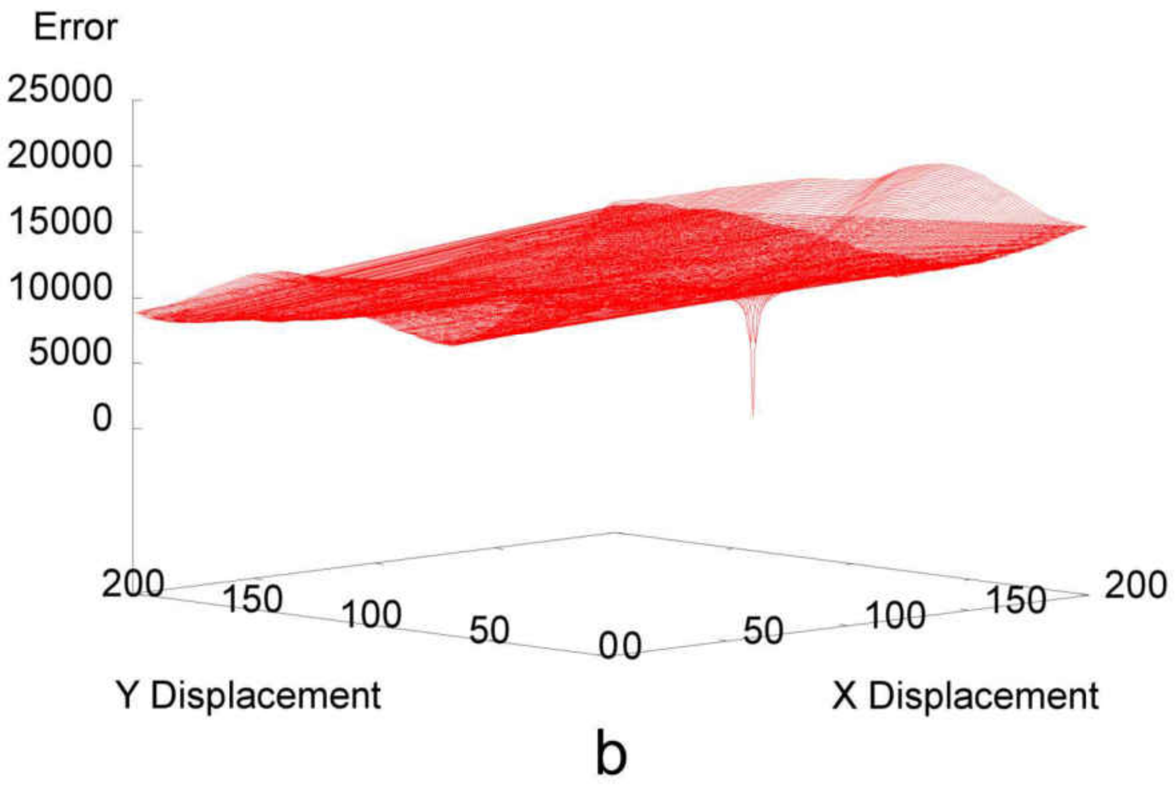


Figure 3.14 - Patch located on region containing the most edges (a) and used as a target to draw the error surface (b)

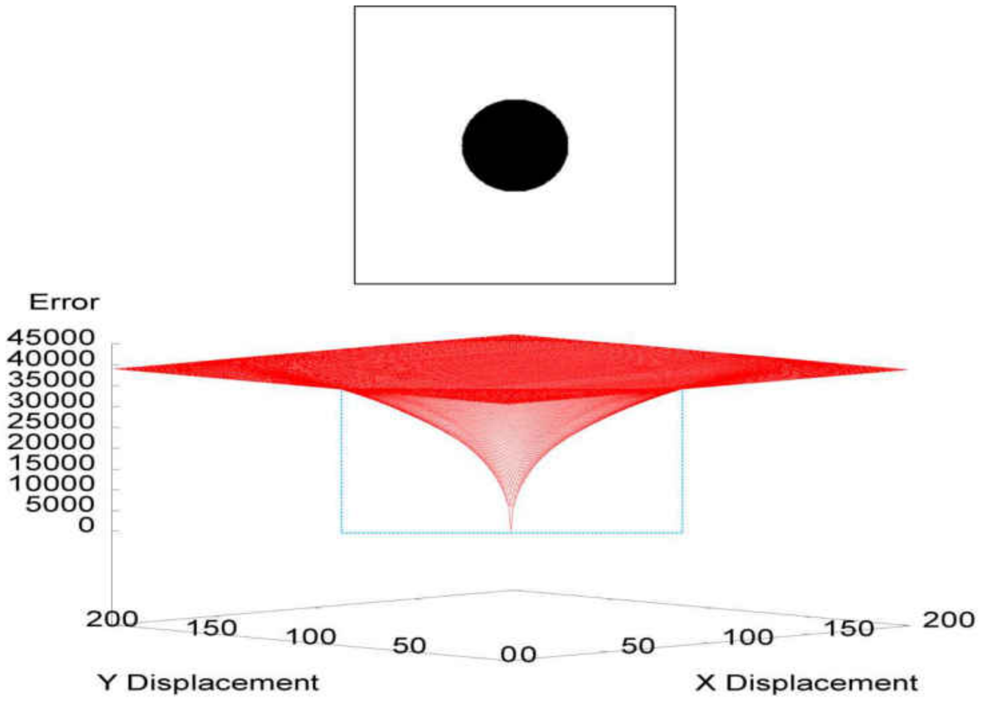
The gradient descending slope in the error surface in Figure 3.14.b looks narrower than some previously shown examples (see Figures 3.10, 3.11 and 3.12). This will cause problems with defining the global minimal point from the majority of the surface's locations.

3.2.1.3 Clear and Fuzzy

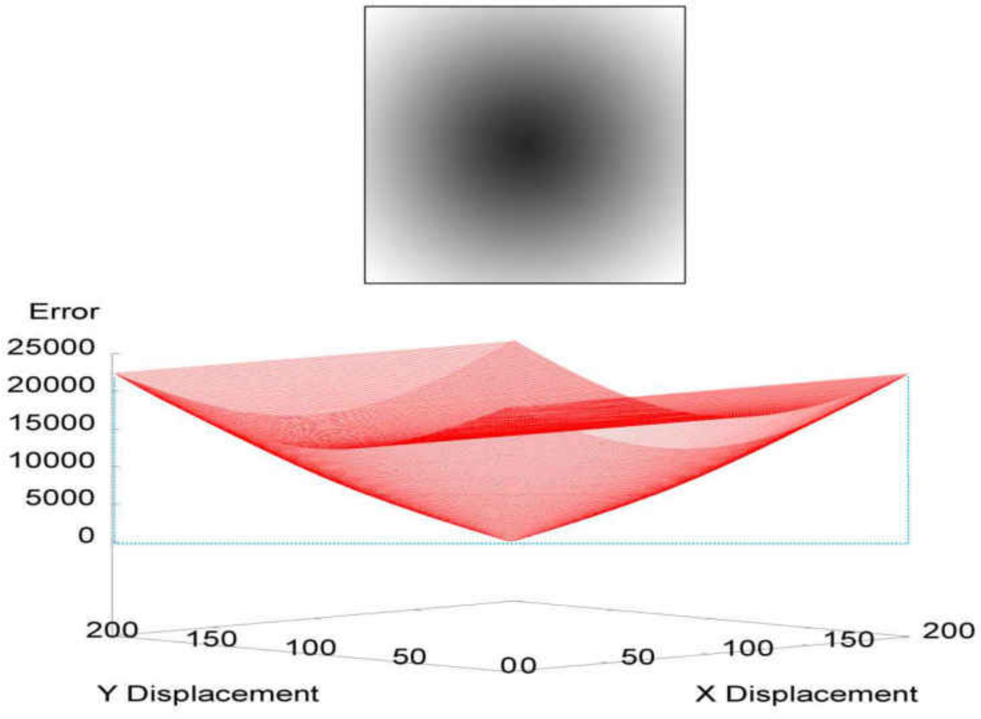
Edges are an important feature in generating the shape of error surfaces. Clear images usually have hard edges, whereas fuzzy ones contain soft edges.

3.2.1.3.1 Artificial Images

Images with a clear resolution will usually have sharp edges and we expect sharp variations on the surface, as patches used for generating error surfaces pass through such image regions. In non-fuzzy images we expect error surface slopes to be steeper than those images with a smoother variation. Figure 3.15 shows a visual comparison between two error surfaces which were generated from clear and fuzzy circles. It shows how the use of sharp edges (i.e. Figure 3.15.a) also generates a surface with a sharp slope. Moreover, the blue dashed boxes in Figure 37 are demonstrating the gradient descend areas for both error surfaces. It shows the located dashed box below the fuzzy image's error surface (see Figure 3.15.b) to be wider than the clear image's error surface. However, to be more precise, we are able to manoeuvre towards the minimal point from any point shown in the Figure 3.15.b error surface but this ability gets restricted in Figure 3.15.a.



a



b

Figure 3.15 - Error surface (a) was generated from image with a sharp edged circle, whereas error surface (b) was generated over an image with a fuzzy circle. Gradient descending areas are shown by dashed blue lines

Both surfaces in Figure 3.15 have well-defined global minima and the minima could be found effectively using simple control systems such as PID. The surface generated from the clear image is steeper than the smoother surface generated by the fuzzy image. Fuzzy images will have more gradual changes which makes them a better choice if we need a smooth error surface for the control system to work on.

3.2.1.3.2 Real World Images

Section 3.2.1.2 represents an experiment on locating the patch over the highest number of clustered edges within the image's space. The aim here is to follow what has been applied in section 3.2.1.3.1 but now on real world images. The process is to blur the image and observe the difference in how error surfaces are generated when the same image is to be blurred and sharp edges are decreased. Figure 3.16.a shows the patch is located over the location which holds the largest number of edges, then the same image is blurred (see 3.16.b) and the same position is used to generate two different error surfaces. The idea here is to observe the difference that it can make to the error surface when the image is blurred.

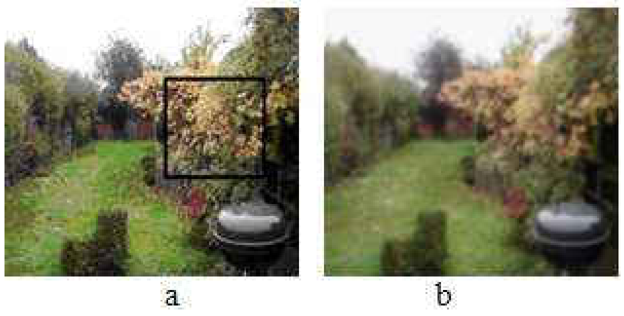
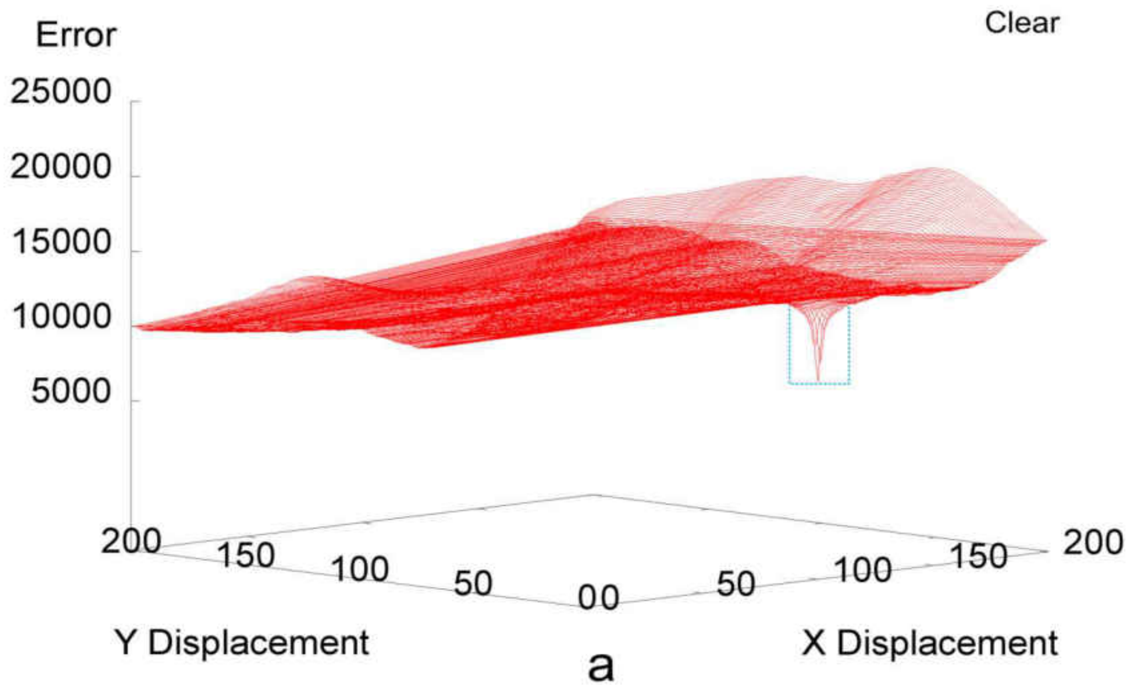


Figure 3.16 - Area with highest number of edges (a) applied over the fuzzy image (b)

Figure 3.17 shows a comparison between the two surfaces obtained with the two images using the patch indicated.



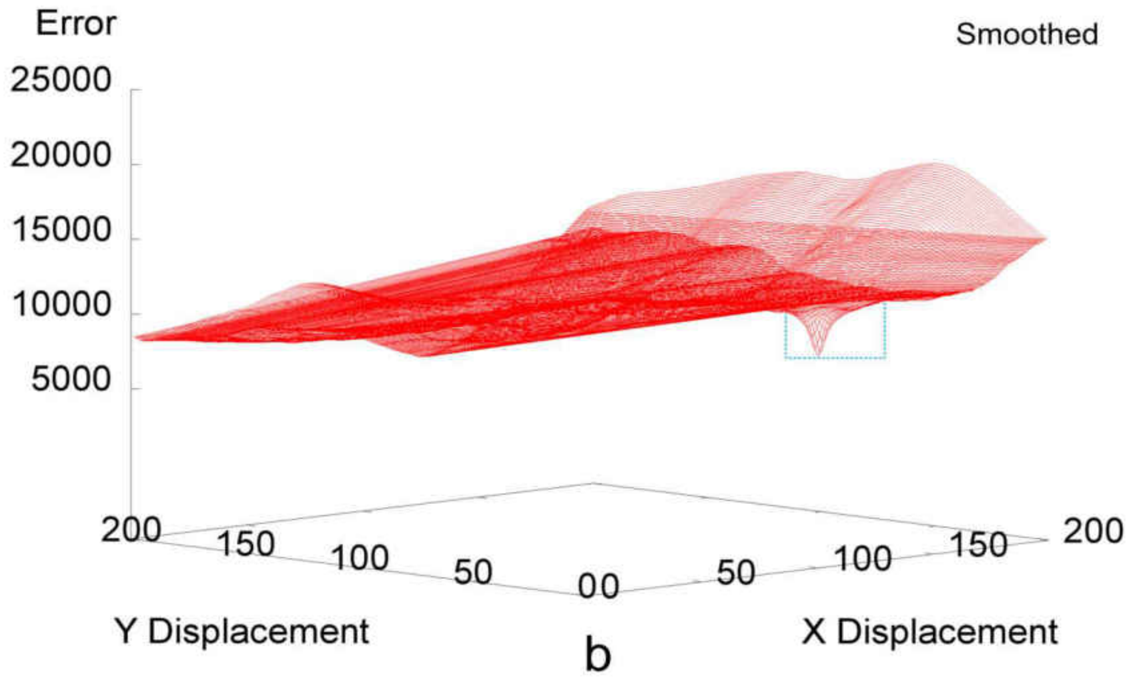
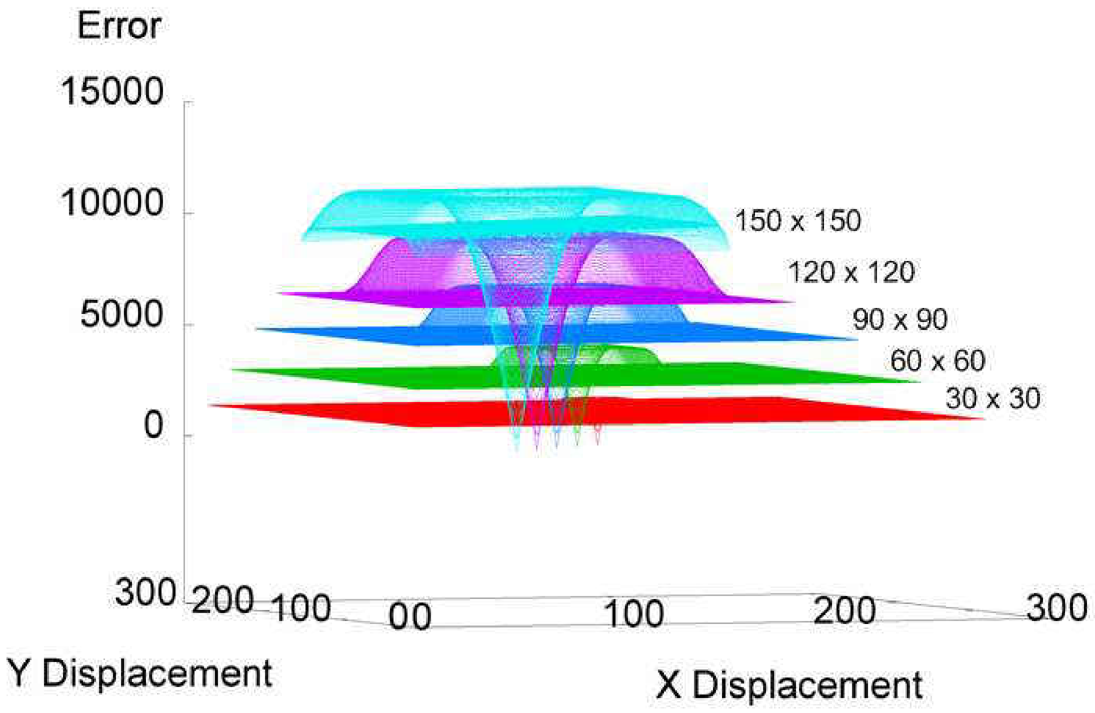


Figure 3.17 - Error surface (a) was generated from garden image with sharp edges, whereas error surface (b) was generated over the fuzzed garden image. Gradient descending areas are shown by dashed blue lines

As in Figure 3.15, in Figure 3.17 the smoothening process over the garden image has also broadened the gradient descending area (shown by the blue dashed box), therefore, we are now aware that the smoothening technique helps to increase the potential gradient descending area on error surfaces.

3.2.1.4 Artificial Shapes

A further experiment examined what artificial shapes produce the best error surface shapes. We use fuzzy blobs of different sizes that gradually change colour from the centre towards the circle's edge. The patch used captures the entire fuzzy blob and some white space surrounding it. The purpose is to investigate which of those blobs produce the best error surface for minimisation. We then use that artificial patch to iterate through the real world image to find the best match between real world features and the artificial shape. This best match patch is then used as the target in the hope that the error surface will be close to what was produced in the artificial examples (Figure 3.15). Figure 3.18 shows the shaded circle in the centre which will be used to find the best error surface shape that can be used for the comparison process. We tested varying sizes for the fuzzy blobs, such as (30×30), (60×60), (90×90), (120×120) and (150×150) pixels in diameter. The main reason is to capture a patch which includes a dark blob in the middle with some white space surrounding it. Figure 3.18 demonstrates those patches with red boxes inside an image space. However, different patch sizes correspond to different blob sizes which will produce different error surfaces with different slopes.



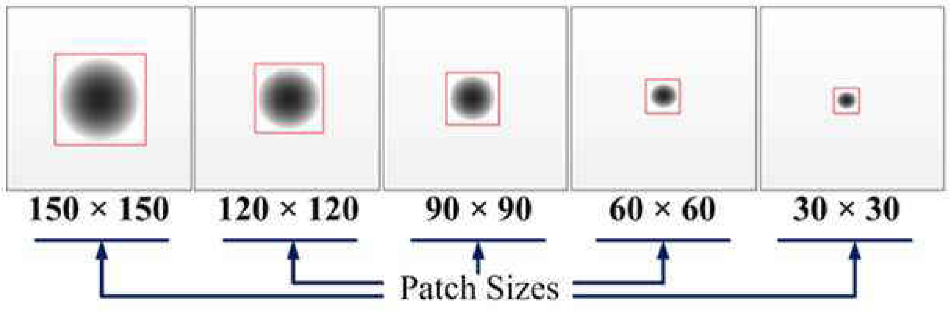


Figure 3.18 - Error surfaces (b) corresponding to different blob sizes (a)

The flattest surface corresponds to the smallest patch with a blob size of *30* × *30* pixels. The other surfaces were produced with blobs of sizes (*60* × *60*), (*90* × *90*), (*120* × *120*) and (*150* × *150*) pixels. Figure 3.18 clearly shows that increasing the blob size also increases the gradient descending area for the generated surface. Such artificial patches can be used to find similar features in images such as the three test images (Garden, Door and Flower). The experiment was only based on one of the RGB colour channels (informal experiments showed no noticeable difference in performance between them). We arbitrarily chose the green channel for later experiments. The boxes indicated in Figure 3.19(a-c) show the patches selected for each colour channel and after applying the comparison process between the artificial patch and the real image over the different channels.

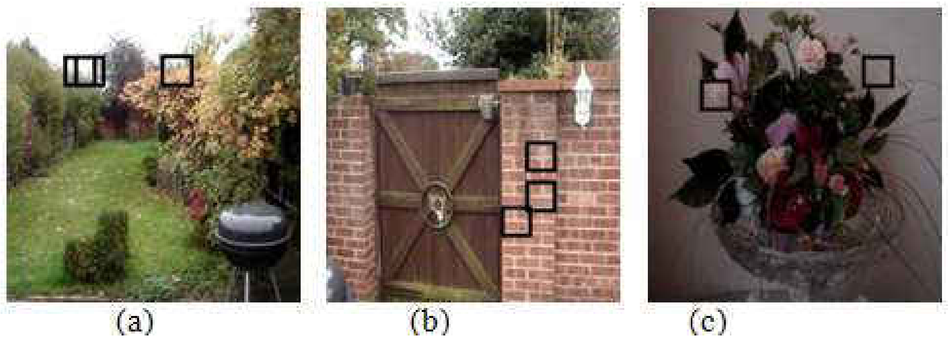


Figure 3.19 - The green channel patch in (a) is located in the top-most left hand box and in (b) it is in the middle between the other two boxes. The green channel box in (c) is located in the top right hand corner where the blue and green patches are overlapping.

The surfaces generated are shown in Figure 3.20 (a-c). These results used the smallest patch size (30×30).

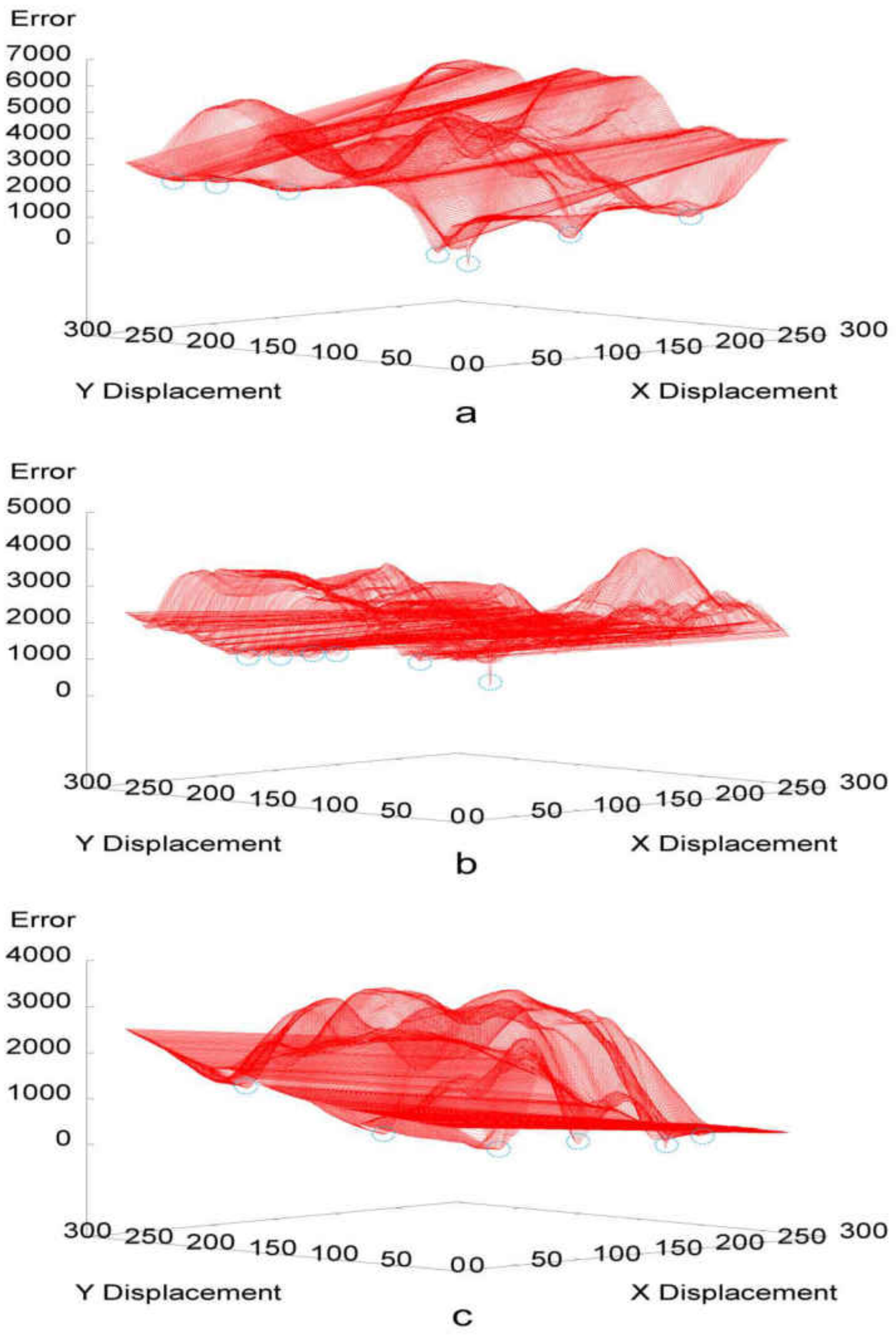


Figure 3.20 - Applying the *30 × 30* artificial patches over the three images. The blue dashed circles show the global minimal point on each surface, whereas the green dashed circles are for existing local minimal points on each surface. (a: Garden, b: Door, c: Flower)

Looking at Figure 3.20, we can observe the global minimal point on each surface which is shown using a blue dashed circle. However, looking at each of the three surfaces, we can still observe a few local minimal points which can mislead the controller to a different location. It is obvious that none of the error surfaces in Figure 3.20 are useful. We also see visually that the green channel patches in all three images were captured in very "bland" areas with little variation which can cause the error surface to have many local minima. We now take the experiment a step further where we repeat the same procedures but with different artificial patch sizes. The green channel boxes drawn in Figures 3.21.a, 3.21.b and 3.21.c indicate the best patch found after applying the comparison between the artificial patch with the real image over that channel only. We have used an artificial patch size of *150 × 150* pixels to find the best patch possible.

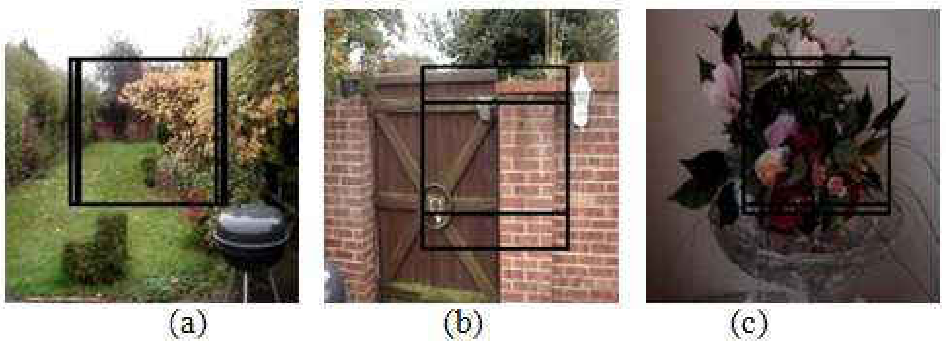
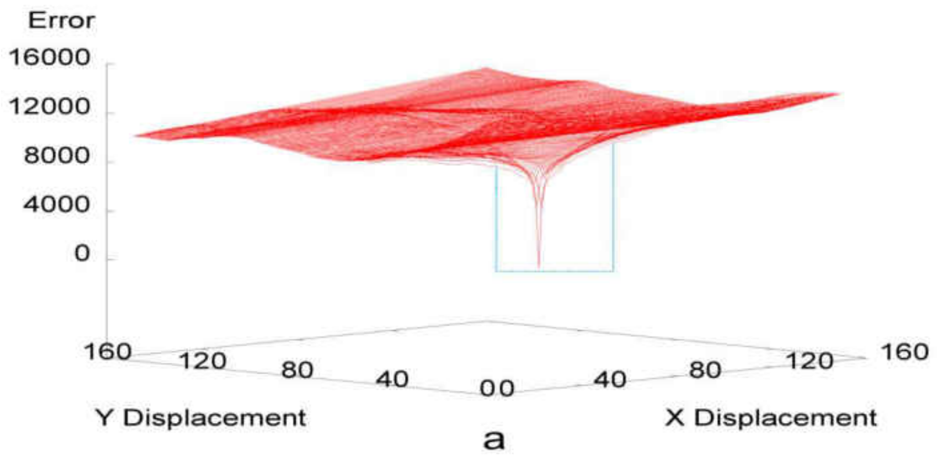
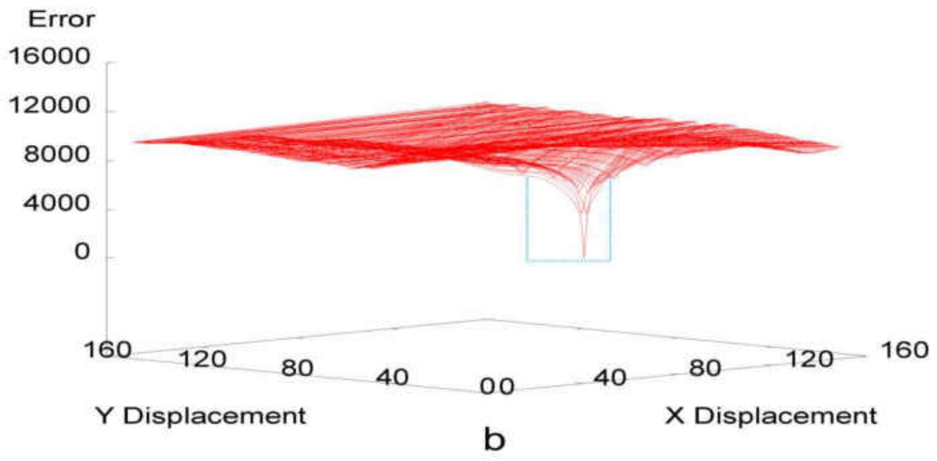


Figure 3.21 - The green and the blue channel boxes in (a) and (b) are mostly overlapping; in (a) it is the left-most box and in (b) it is the top-most box. The green channel box in (c) is overlapping with the red channel box and is the top-most box in the image.

We now apply the comparison with different artificial patch sizes to see how the error surface can differ when capturing a bigger patch and apply the process using a large patch size (see Figure 3.22).





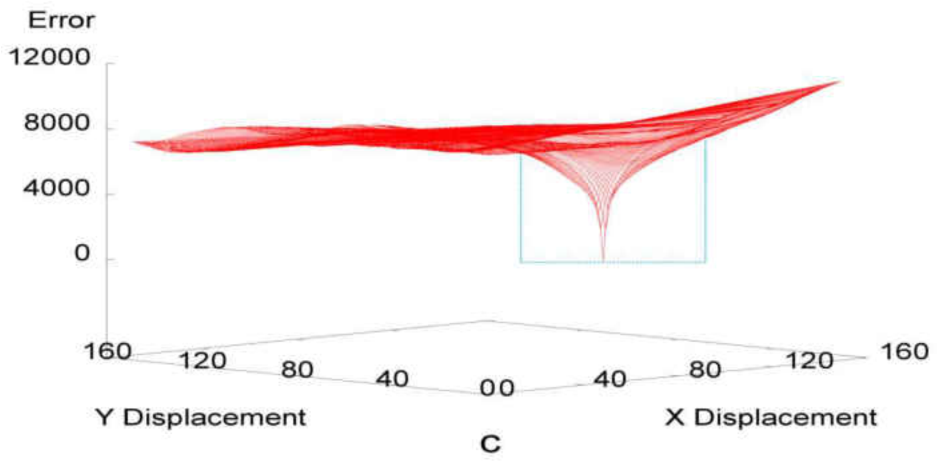


Figure 3.22 - Applying the *150 × 150* artificial patches over the three images. The blue dashed boxes are demonstrating the most gradient descending area for each surface (a: Garden, b: Door, c: Flower).

The difference between the patch sizes is clear and dramatic. For instance, if we compare the Door error surface (Figure 3.20.b) generated by the *30 × 30* patch with the error surface generated by the *150 × 150* patch (Figure 3.22.b), we can see how the large patch removed a lot of noise on the surface and made it much smoother and easier for the controller to navigate. A similar evaluation also applies for the other two error surfaces where the patch's enlargement made a significant difference for both the Garden and Flower images.

3.2.1.5 Whole Image Sampling Strategies

We cannot in general predict what image properties we will encounter and where to apply which strategy to capture the best patch possible. We therefore applied some “whole image strategies” which ensure that no matter what features and what colours the image includes, we still target the entire image and use the entire image as a patch. Using all the pixels requires a prohibitively large amount of processing time and therefore, we applied different ways of capturing a limited but distributed number of pixels around the image. There are two ways to distribute the pixels in the image, one is static positioning and the other is random positioning.

3.2.1.5.1 Static Positioning

This strategy uses most of the image space to capture a variety of pixels from the entire image. The static positioning method uses a fixed grid to select which pixels are targeted. There are many ways of targeting the pixels but we attempted to define a layout to ensure that on all translations some sampled pixels will overlap. Selecting based on a regular grid does not achieve this: if we sample every *n*pixels we will not get any overlap if we displace the image by *(n-1)* pixels.

We resolved this issue by defining a simple way to guarantee some overlap on every movement and therefore we made the selection process take place every 10, then 9, then 8, etc., down to a spacing of 1. This process of selecting the pixels is repeated right across the image. Figure 3.23 shows the selected pixels in the *Static Positioning* strategy. We will be using the above indicated locations as our patch for the comparison process.

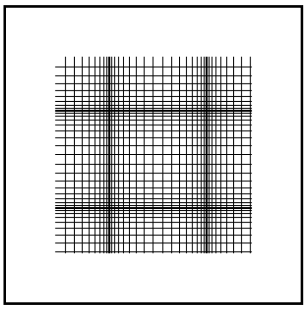
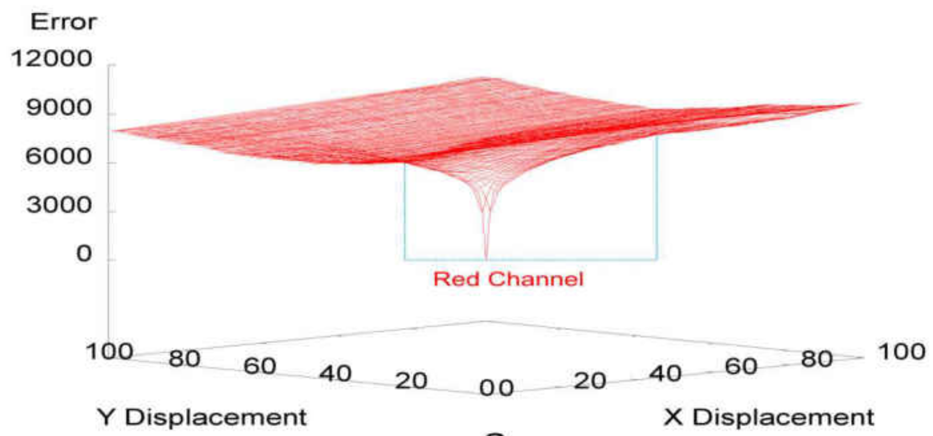
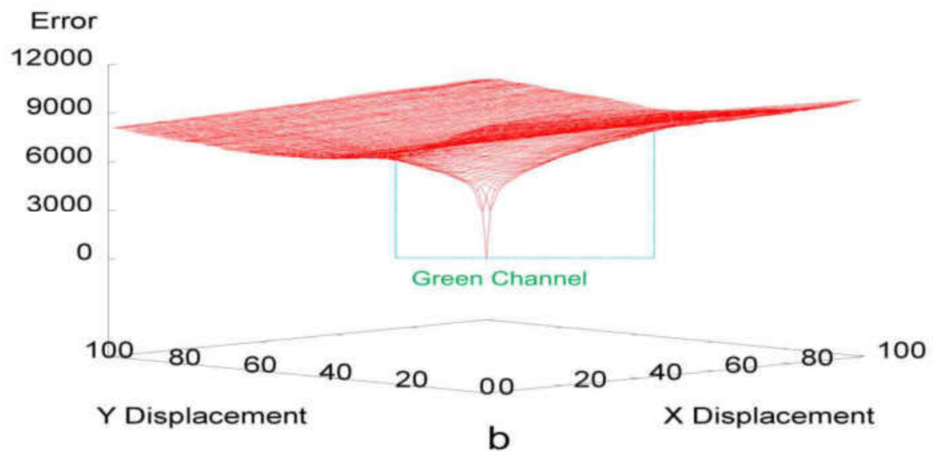


Figure 3.23 - Locations of the Static Positioning strategy

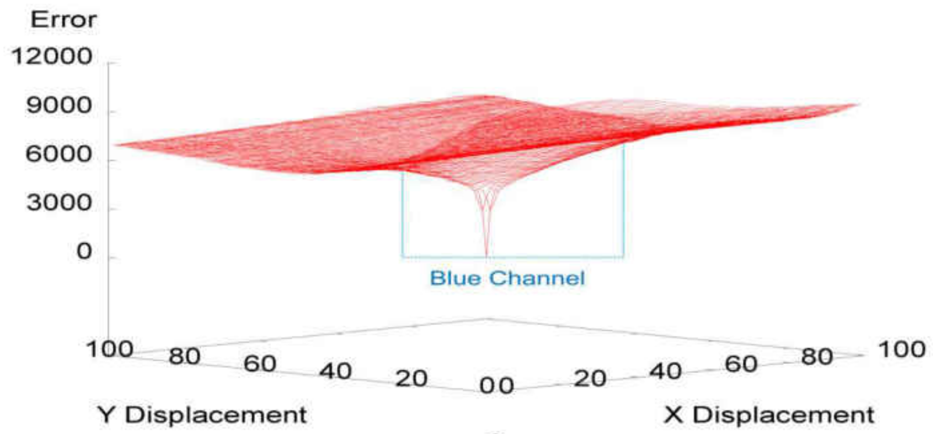
Figures 3.24, 3.25 and 3.26 are demonstrating the comparison process for each colour channel separately. The drawn blue dashed box below the gradient descending area of each surface shows that there is no such difference in the performance of each of the three colour channels over any of the three images (Garden, Door and Flower images). As a consequence, we then decided to use only a single colour channel for the comparison process; therefore, we chose a green channel for no particular reason.



a



b



c

Figure 3.24 - Three error surfaces for the garden image were generated to represent the Red, Green and Blue channels separately. The blue dashed boxes are demonstrating the most gradient descending area for each surface (a: Garden, b: Door, c: Flower).

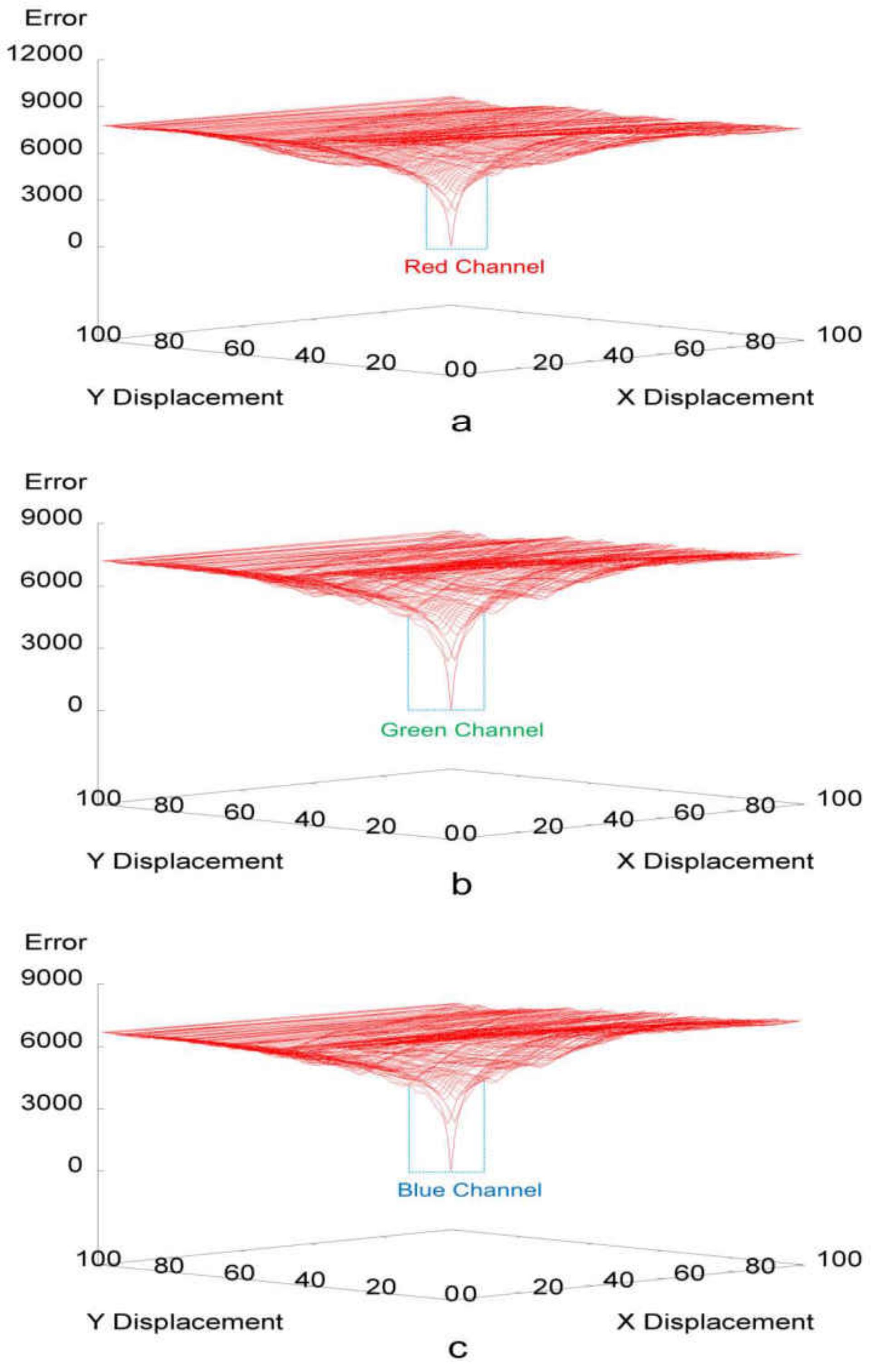


Figure 3.25 - Three error surfaces for Door image were generated to represent the Red, Green and Blue channels separately. The blue dashed boxes are demonstrating the most gradient descending area for each surface (a: Garden, b: Door, c: Flower).

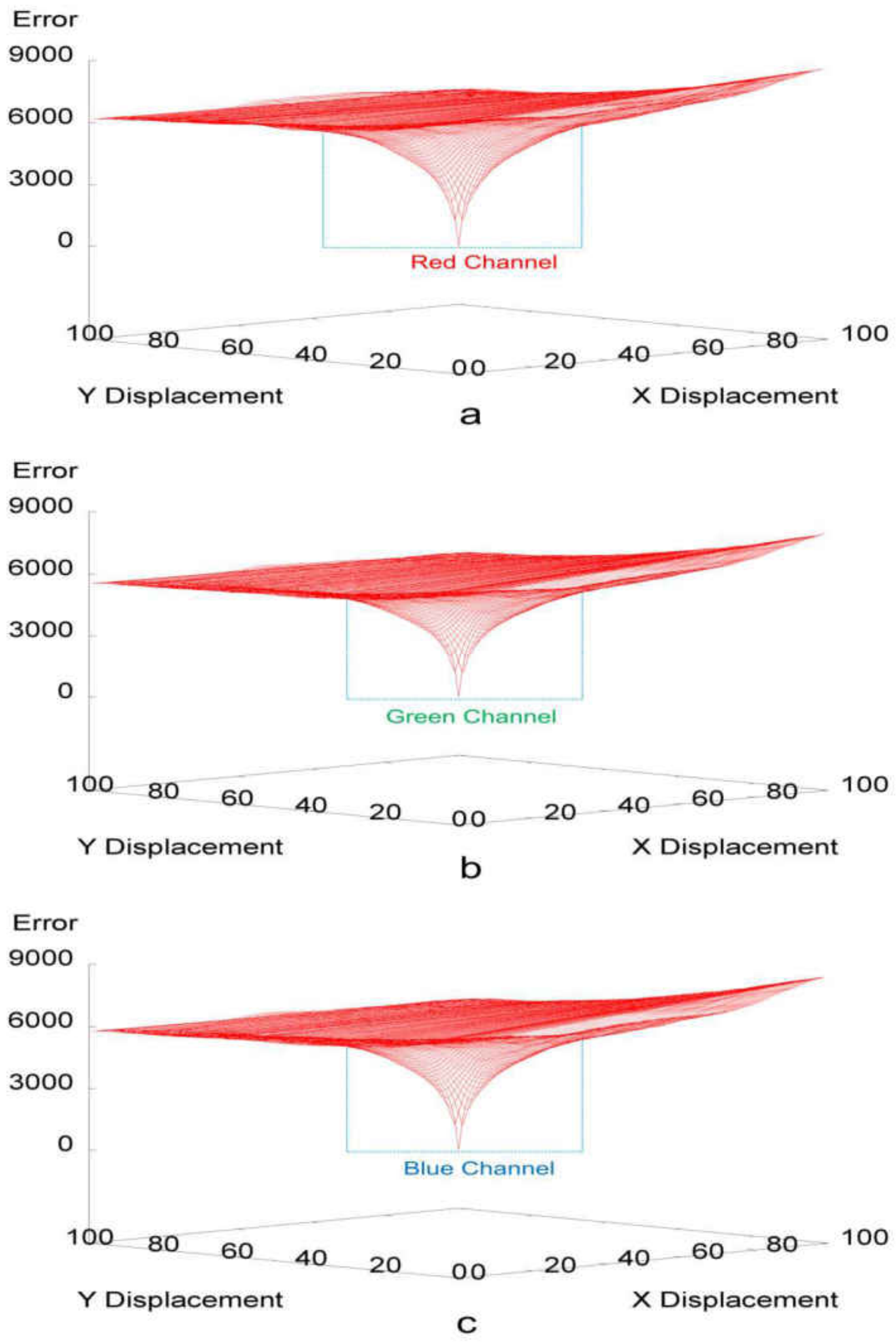


Figure 3.26 - Three error surfaces for the flower image were generated to represent the Red, Green and Blue channels separately. The blue dashed boxes are demonstrating the most gradient descending area for each surface (a: Garden, b: Door, c: Flower).

Figure 3.27 shows error surfaces for the Garden, Door and Flower images using static positioning over the green channel only.

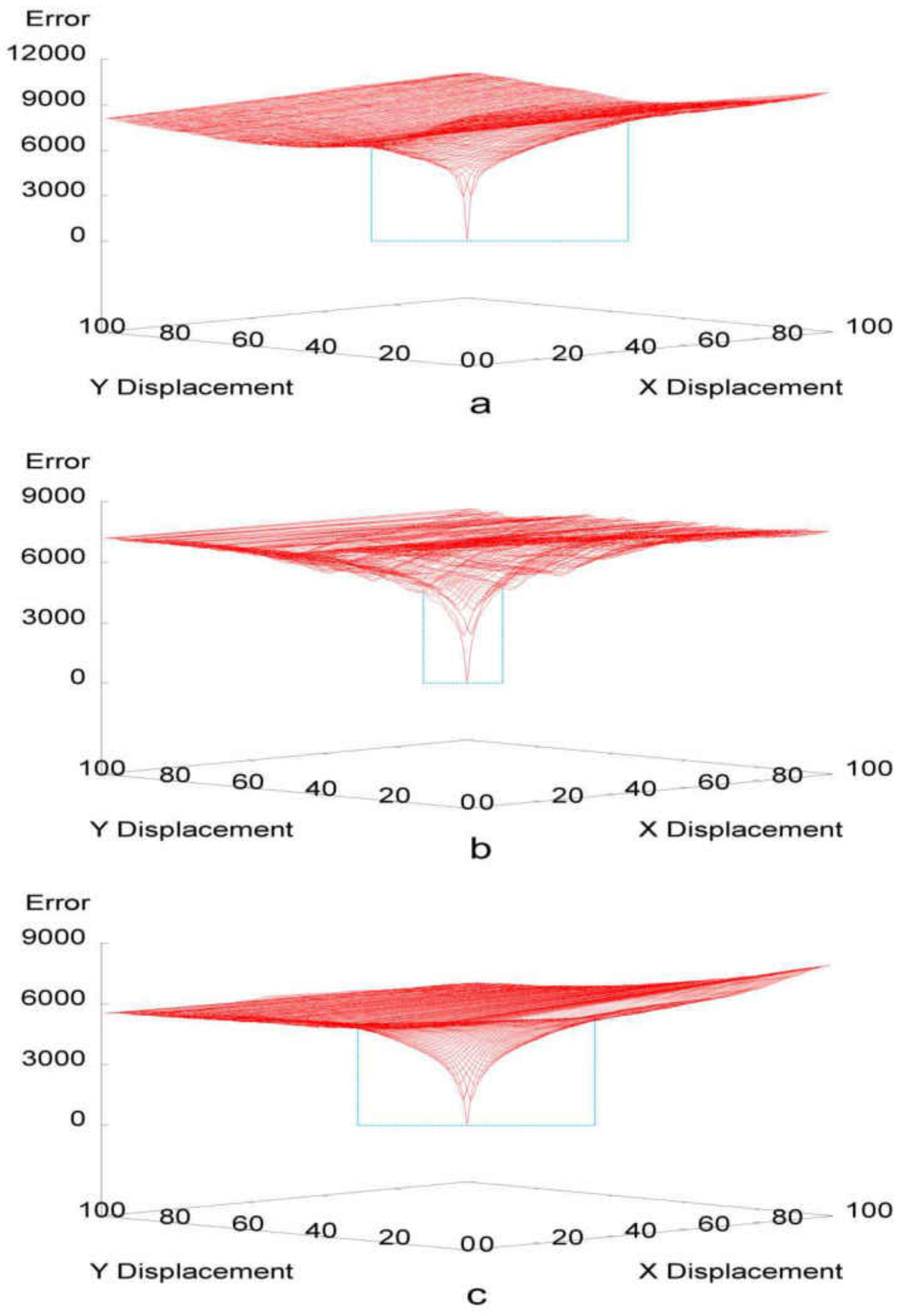


Figure 3.27 - Error surfaces for different images after applying the *Static Positioning* strategy. The blue dashed boxes are demonstrating the most gradient descending area for each surface (a: Garden, b: Door, c: Flower).

It is clear that all three error surfaces have a smooth slope down towards their minimum and the noise and local minimal are almost completely eliminated (although local minima due to the brickwork texture are apparent in Figure 3.27.b, which is potentially valuable from the controller's point of view.

3.2.1.5.2 Random Positioning

Random Positioning of pixels is another technique which we used to target the entire image rather than positioning a patch(s). Figure 3.28 shows the uniform random distribution of locations used in this strategy. To implement this randomness, we have designed a class in C++ **language.** The implementation of the Random class is based on Donald E. Knuth's subtractive random number generator algorithm [134].

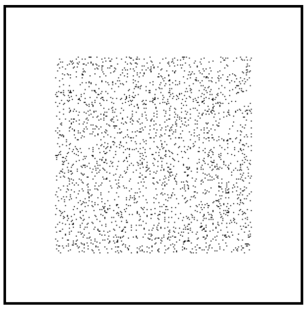
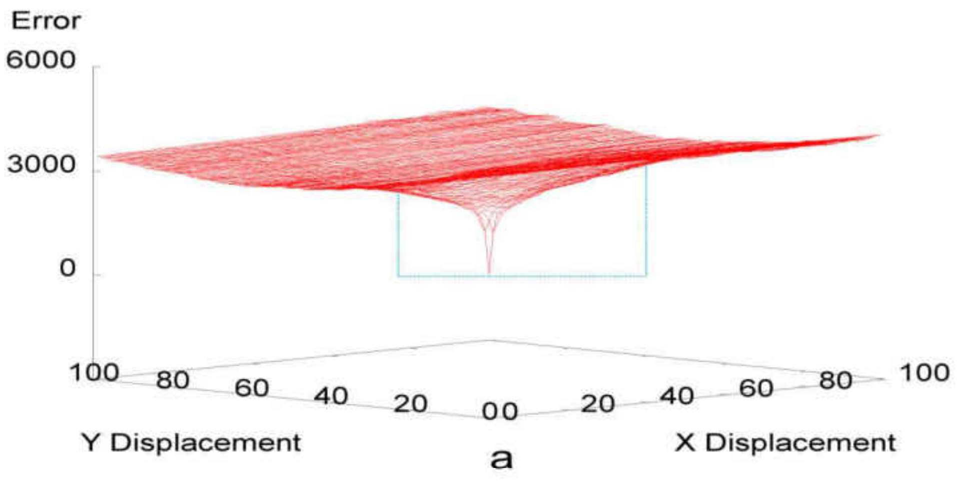
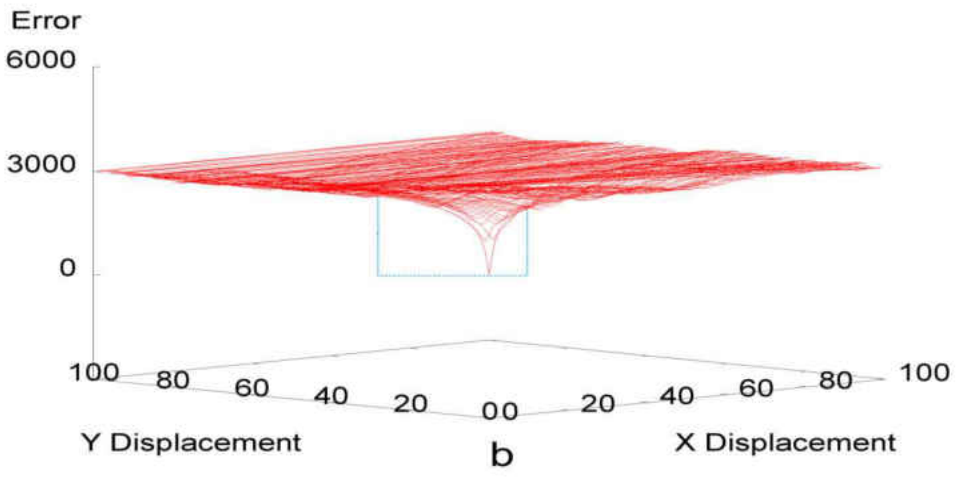


Figure 3.28 - Distributions of the pixel locations using the *Random Positioning*

In this approach we also ran the comparison process over each of the colour's channels separately to see if there was any difference amongst any of the produced surfaces. As in previous cases, we have chosen the green channel to base our experiments on as there was no noticeable difference.





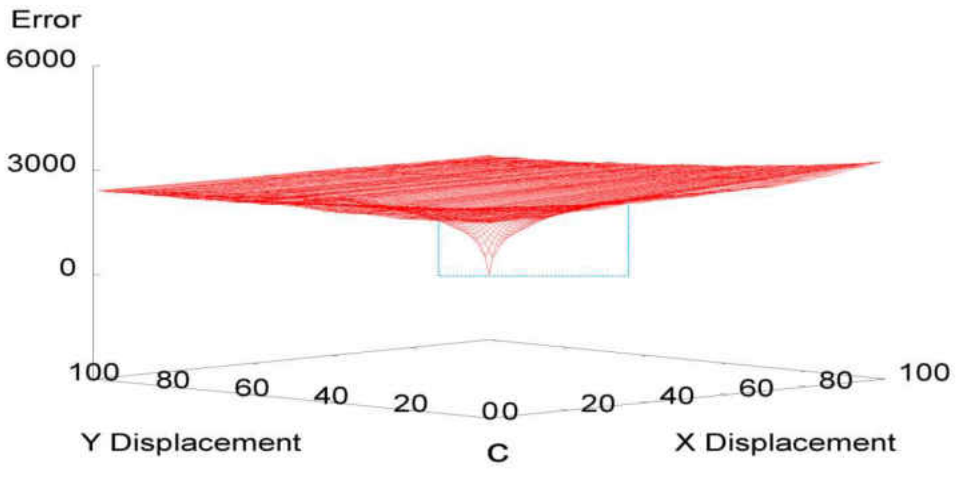


Figure 3.29 - Error surfaces for different images after applying the Random Positioning strategy. The blue dashed boxes are demonstrating the most gradient descending area for each surface (a: Garden, b: Door, c: Flower).

Figure 3.29 shows the error surfaces generated using the random positioning strategy based on the green channel only. The large gradient descending areas are represented by the drawn blue dashed boxes below the gradient descending areas of the Static and Random surfaces (see Figures 3.27 and 3.29).

Now, seeing how well the Error Surfaces were generated using both methods in the Whole Sampling, we now need to make the final decision about which one of the two methods are best to be used (Static or Random). To achieve this, we would run a simple control algorithm to simulate the tracking process and see which one of the designed methods will be performing better.

In order to meaningfully assess the utility of the error surfaces we need a control system that can use them to find the minimum. There are a variety of control algorithms that can be used to control systems such as that presented here. PID (Proportional, Integral, and Derivative) [3] is one of the simplest and best understood and a slightly modified proportional controller will be used in this work.

Our "P" algorithm enhances the controller movement by increasing the P gain iteratively when the control system fails to generate a movement. Figure 3.30 shows the two scenarios of when the P value needs to be increased or not. The increase of the P value is a simple low-overhead heuristic to avoid getting stuck on the flat regions of the error surface. The P gain is reset to 1 after each movement. This algorithm is not intended to be used in a final control system, but is simply used to assess the error surfaces generated in this work.

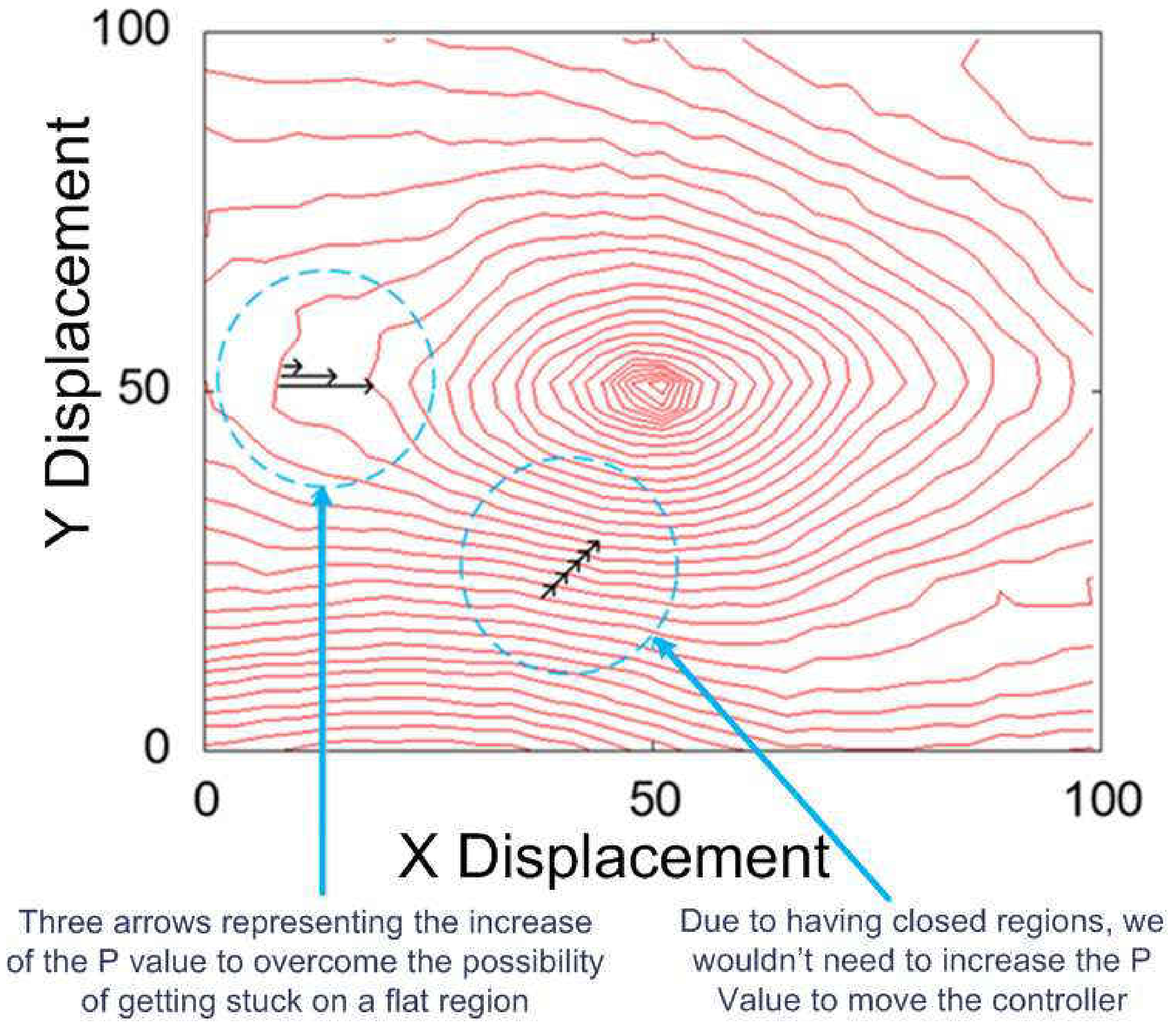


Figure 3.30 – Representation of two scenarios where one uses a fixed P value and the other increases the P value gradually to overcome the large distance between the error contours' levels

The next stage is to test how well the controller navigates over those generated surfaces using both the Static and Random Methods. We first start to run the P controller over the three generated Error Surfaces using the Static Method. Figure 3.27 demonstrates the three generated Error Surfaces which will be used to testify the controller's performance. Running the control algorithm described above from all starting points on each error surface will give us the ultimate result which verifies from what parts of the surface we can reach the target and get ourselves back to the original location. Figure 3.31 is produced from the Garden's green channel error surface. Figure 3.31.a shows a number of dots on the contour map which indicate the locations from which the minimal point was successfully reached using the simple control algorithm. Figure 3.31.b shows the distribution of the number of steps required (Y axis) to reach the global minimum from all the successfully minimised starting locations (X axis).

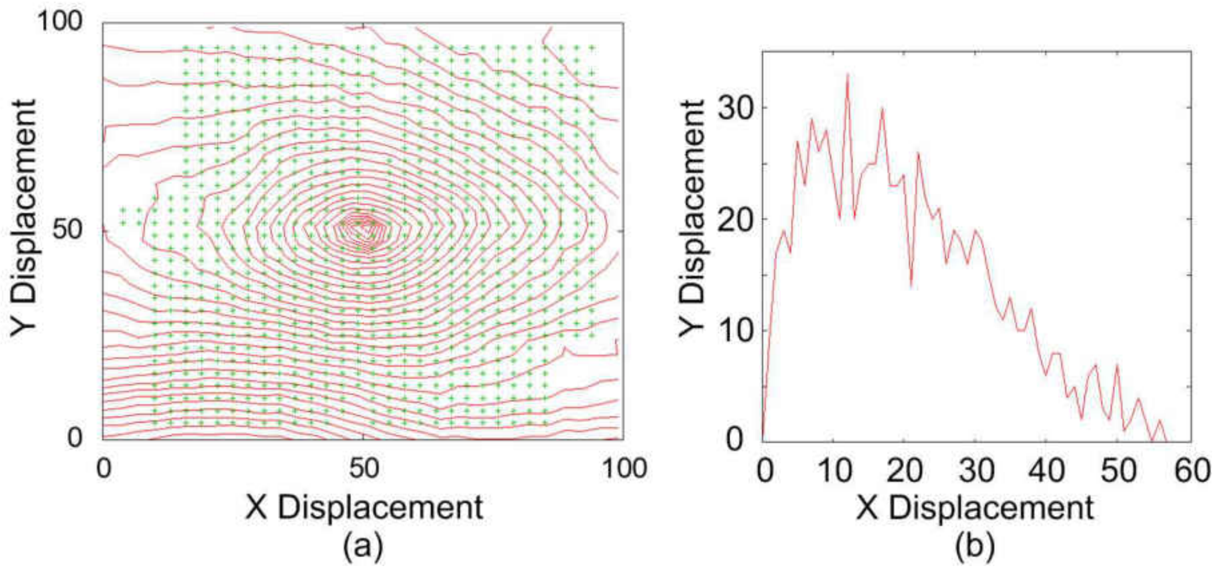


Figure 3.31 - Successful areas on the contour map for the Garden image

P controller will use all available locations on the surface as a starting point towards the minimal point. However, if P controller reaches the minimal point, its starting location will be considered as a successful location. For example, in Figure 3.31, P controller could reach the minimal point from 77% of starting locations; therefore here we consider that the success rate in Figure 3.31 is 77%. Figure 3.32 shows the number and locations of the successful starting points on the error surface for the Flower image. It shows that 73% of the entire image's locations were successful.

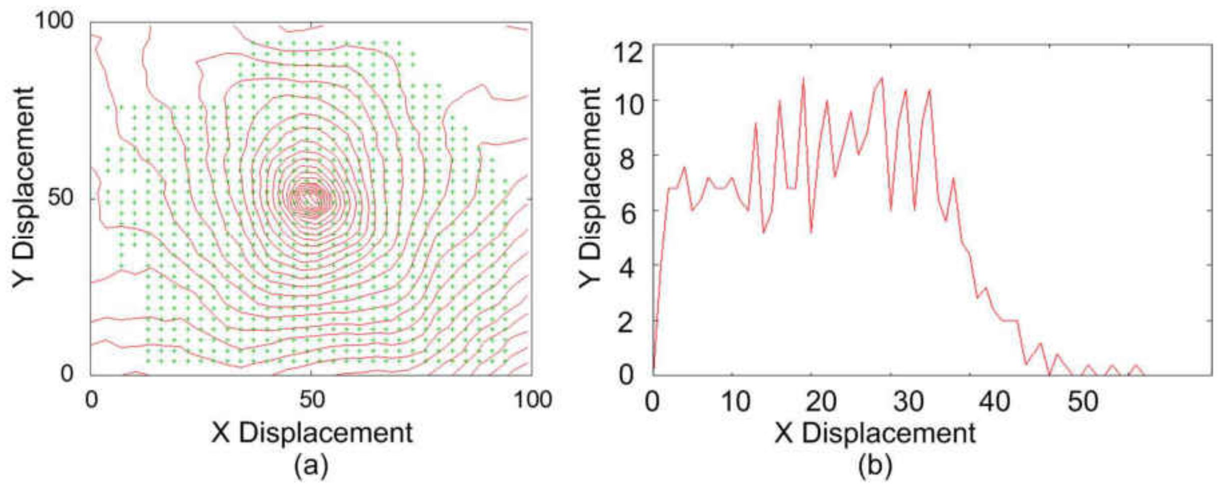


Figure 3.32 - Successful areas on the contour map for the Flower image

Figure 3.33 shows the same data for the Door image, for which 20% of the starting locations succeeded.

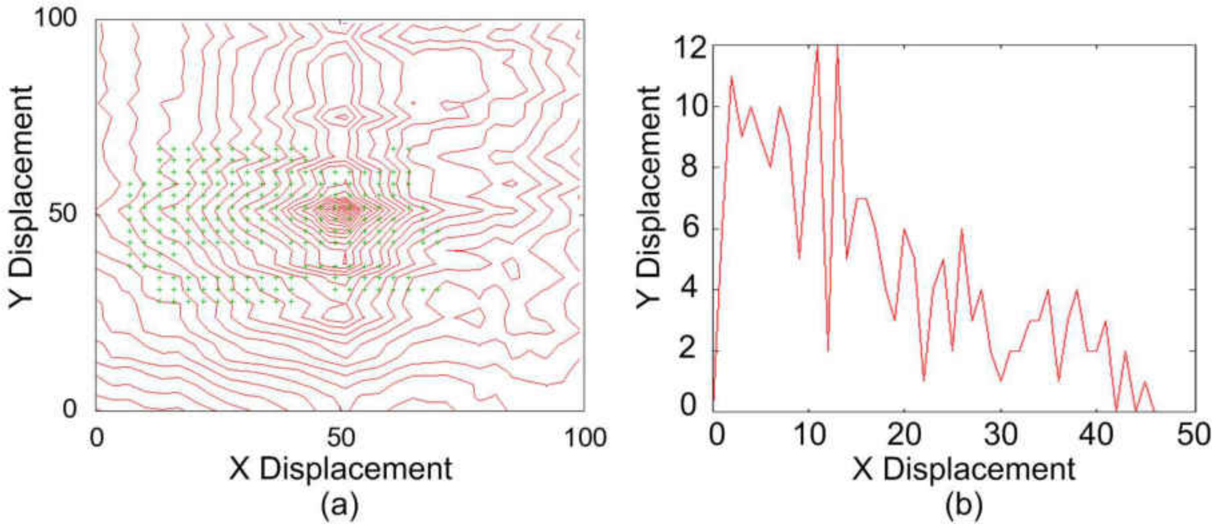


Figure 3.33 - Successful areas on the contour map for the Door image

This poor success of the Door image is due to the homogeneous areas and repeating patterns in the image.

Now, the same procedures will be followed using the Error Surfaces generated by the Random Method. Starting with the Garden Image's Error Surface, Figure 3.34.a is a contour map for the

Garden image with the indication of the successful points and Figure 3.34.b is a graph showing how many points took how many steps in order to achieve the process.

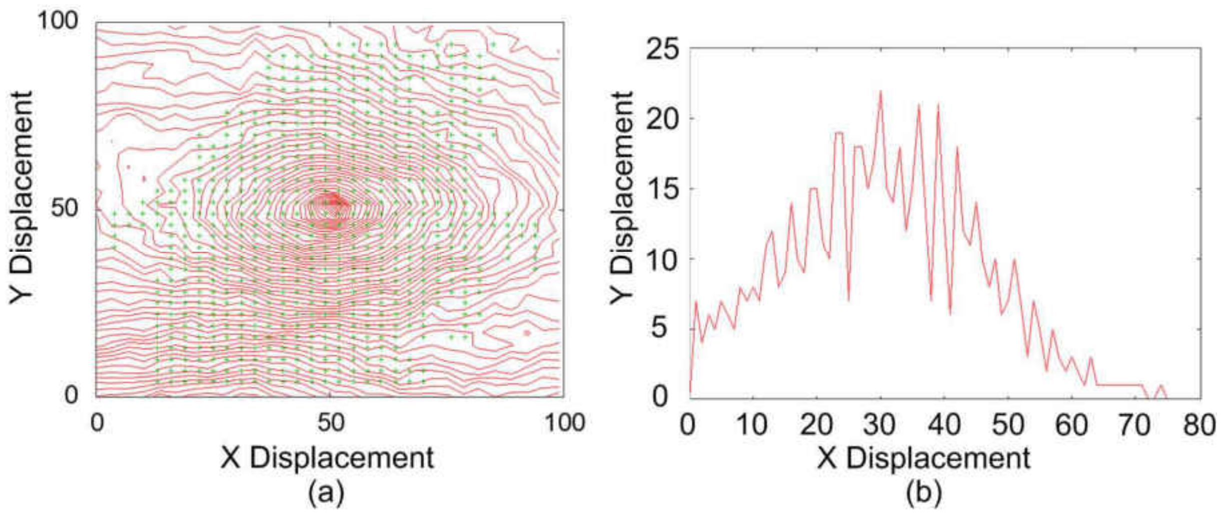


Figure 3.34 - Successful areas on the contour map over the Garden image

In this case, using only the green channel, 59% of the starting locations succeeded using the *Random Positioning* strategy. Figure 3.35.a shows the contour map for the Flower image with the indicated points showing the successful area, which is 52% of the image locations.

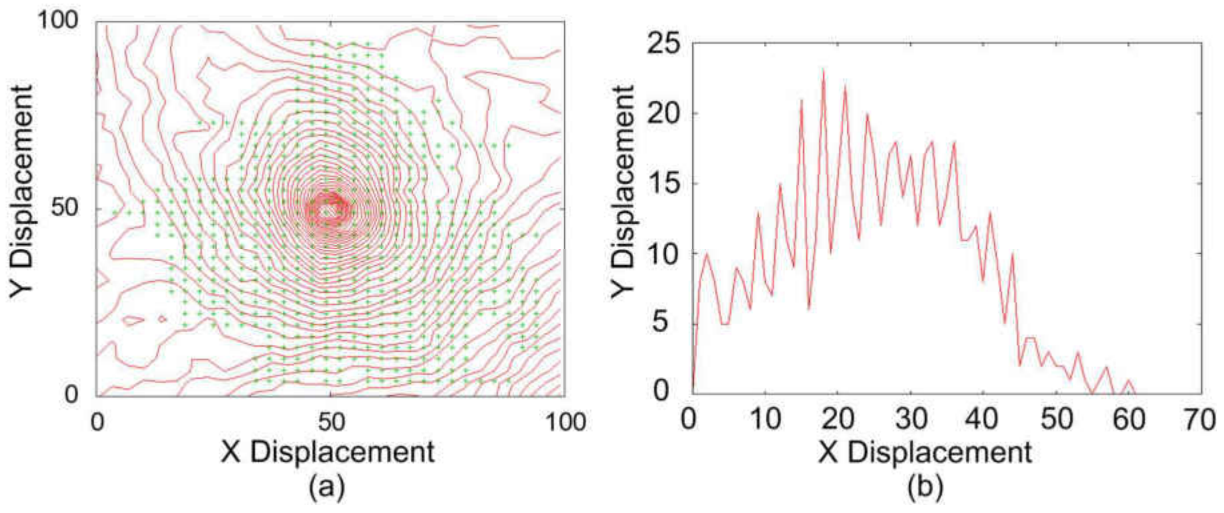


Figure 3.35 - Successful areas on the contour map for the Flower image

Finally we apply the same technique to the Door image. Figure 3.36.a shows the successful area covering 24% of the image: a slight improvement over the *Static Positioning*.

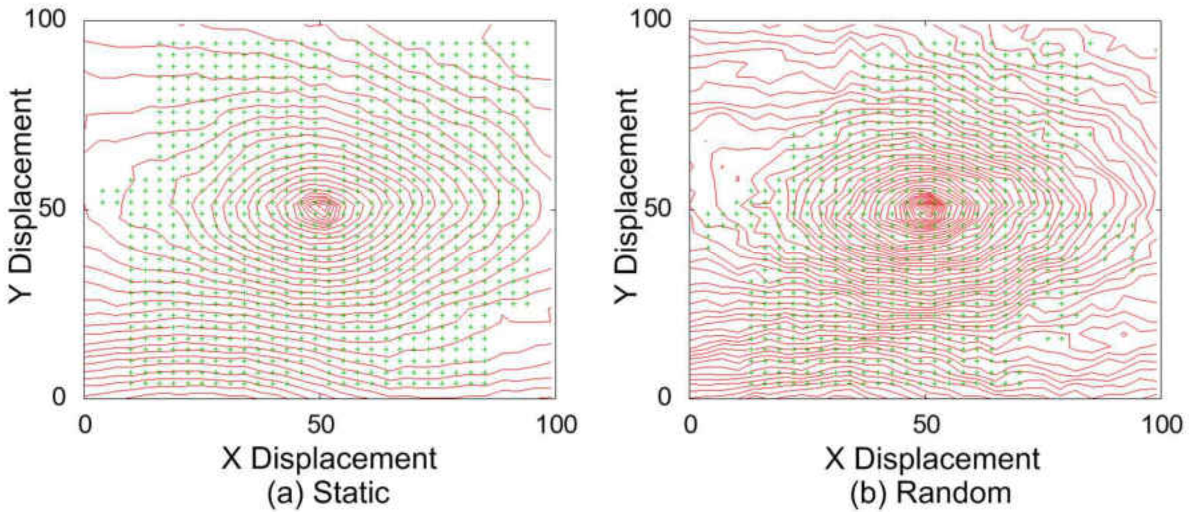


Figure 3.36 - Successful areas on the contour map for the Door image

3.3 Conclusion

The purpose of choosing different strategies was to implement a solution for the best patch found within the image. The fixed locations strategies only perform well if the patches happen to capture some good features in order to build the error surface but this is unacceptably dependent on the properties of each individual image. The patch size is also a major consideration in the shapes of the error surfaces: larger patches tend to produce better error surfaces and the effects of "noise" decrease. The strategies using edge detection and artificial patches also performed unreliably in general.

There is a dramatic change in performance between *patch positioning* and *whole image sampling* strategies. For example, Figure 3.37.a was generated by applying the *60×60 Artificial Patch* to find the best positioning before applying the comparison process and the surface in Figure 3.37.b was generated by applying the *Random Positioning* strategy. The random strategy uses fewer pixels for the comparison process and the difference between the two is clear: the random surface is smooth, evenly sloping and has a well-defined single global minimum, whereas the artificial patch surface is noisy and has multiple local minima. Both static and random positioning strategies perform well and quite similarly by targeting pixels distributed over the entire image.

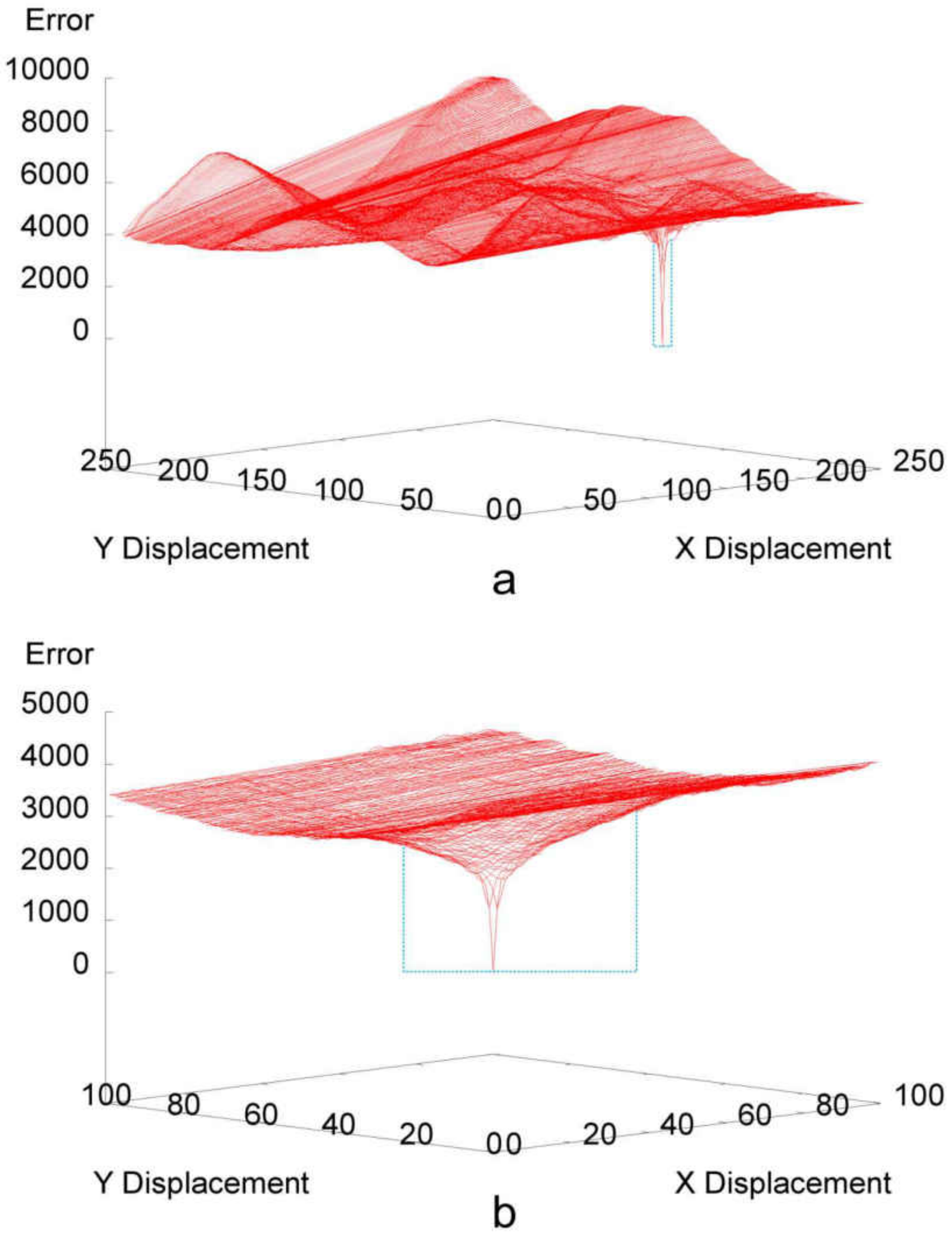


Figure 3.37 - Patch positioning (a) and Random positioning (b) surfaces. The blue dashed boxes are demonstrating the most gradient descending area for each surface.

Both whole image sampling strategies are effective, although in these experiments more pixels were sampled in the static sampling strategy. Despite the large number of pixels, the error surface contains a number of regions from which our simple controller was unable to minimise. This may be partly due to the simplistic nature of the controller, but in general the randomly positioned pixels yield steeper and deeper global minima: compare Figures 3.38.a and 3.38.b.

In general the *Random Positioning* strategy does not suffer from regions in which our controller was unable to minimize and therefore seems to be a better solution.

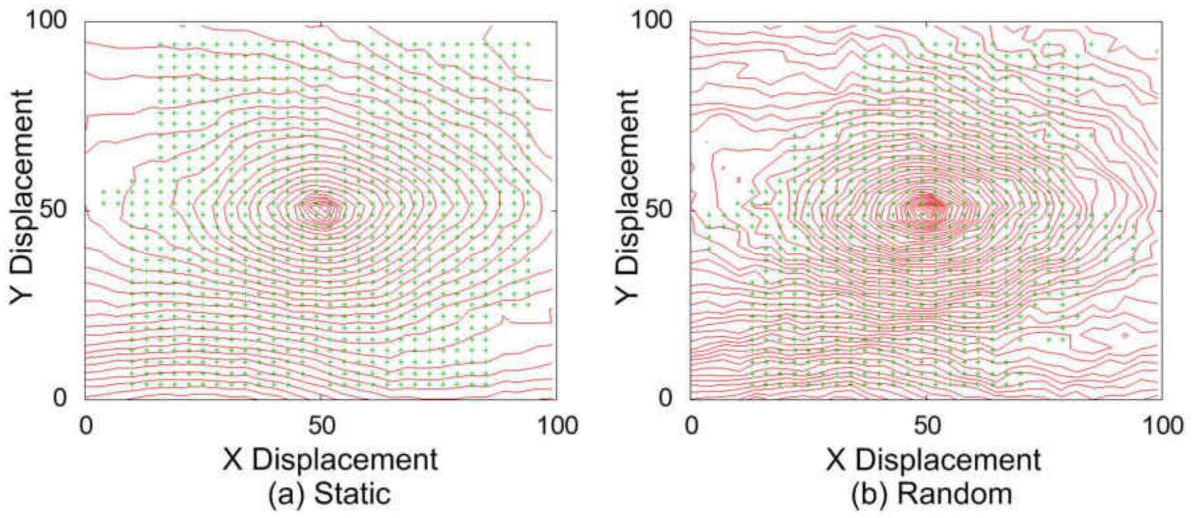


Figure 3.38 - Successful areas on Static (a) and Random Positioning (b)

The minimisation method used on the surfaces generated by this work was essentially a P controller moving the target location iteratively across the surface. The controller only used information from the local slope of the error surface which was assessed using three points to obtain the direction of the slope; this minimises the computational load by only requiring the sample pixels to be compared three times between each movement of the actuators. A summary of the numbers of pixels sampled to generate some error surfaces can be seen in Table 3.1.

Table 3.1 - Number of pixels sampled for various surfaces

Figure		
3.9 (a,b,c)	10,000 (*Central* + *Individuals*)	10,580 (*Merge*)
3.14 (a)	10,000 (edge detection)	
3.15	10,000 (artificial patches)	
3.19	900 (small artificial patches)	
3.21	22,500 (large artificial patches)	
3.23	14,600 (static sampling)	
3.28	2,500 (random sampling)	

Other than strategies, there are few properties such as surface's roughness and depth of the hole in centre can have major affect on the controller's navigation process. This means, we may have large descending area but if the surface's roughness is high, it results to create many local minima around the global minima which would interfere on the controller's navigation process and may lead it to stuck in one of the local minima rather than navigate towards the global minima.

However, the shape of the hole in centre can also affect the ultimate process. For example, if the hole has a sharp ascending and descending then, it would be difficult for the controller to settle on the global minima. On the other hand, if the hole has a smooth ascending and descending then, the controller may also get unsettled condition by move back and forth around the global minima. It's difficult to setup a technique which can quantify the graph but instead, we used many appearance comparison strategies to compare the generated graphs visually by drawing rectangle around the most ascending and descending area. The drawn rectangle should also consider the roughness of the surface as well. This would visually demonstrate the range of potentiality each graph can have when compared against some other generated graphs.

The ultimate conclusion behind all the patch positioning strategies and the applied experiments is that wider distributions of pixels will capture more of the variation in real images and therefore will produce better error surfaces and less noise. In order to achieve this some form of whole image sampling is the best technique. This leads to successful navigation on the error surface generated by the entire image and does not depend on a specific feature, shape, colour, etc. To design a successful visual stabilisation control system for our Intelligent Kite Aerial Photography Platform (iKAPP) system we need a strategy which reliably generates a good error surface where the controller will have the opportunity to get us back to our target from most positions in most images. In addition, the processing load is important as the platform uses a fairly low specification: 1GHz processor with 512Mb of RAM. Fast processing will allow us to process images at a higher frequency and therefore move the actuators at a higher frequency which will improve the stabilisation performance. Every error surface has a certain region within which the controller can succeed and if the kite system encounters a lot of turbulence then more images and faster processing will help to keep the camera within this region.

Table 3.2 presents a comparison between Static and Random strategies. It clearly shows how Static strategy has achieved higher success rates over two images (Garden and Flower) but achieved less over the Door image. However, success rate of Random strategy in Garden and Flower images is over 50%.

Table 3.2 – Success rate comparison between Static and Random strategies

Image	Strategy	Sampled Pixels	Success Rate
Garden	Static	14,600	77%
	Random	2,500	59%
Flower	Static	14,600	73%
	Random	2,500	52%
Door	Static	14,600	20%
	Random	2,500	24%

When this success rate is balanced against the sampled pixels in each strategy then, Random strategy seems to have more potential from the processing speed point of view over forthcoming experiments.

When we refer back to our question in chapter 1, Random is a good strategy to be used for having a fast processing method to identify the appropriate regions for image-based stabilisation. However, the next chapter will use this strategy in real time process.

Chapter 4

Design Methodologies

4.1 Preface

After seeing how the random strategy was chosen to be our best methodology for pixel selection, we now need to know how to choose the size and the location of our patch.

Patch Size Selection (PSS) and Slice Selection Process (SSP) are the two main sections included here which both describe the entire Patch Selection Process (PSP). However, beforehand, we will be applying some calibration processes before proceeding to further stages on PSS and SSP.

4.2 Calibration Process

We will first need to set up the platform in the laboratory to act as it will act in the real world environment. Figure 4.1 demonstrates how the platform was hung using four strong threads which are used to attach the platform to the laboratory's ceiling. However, the ground distance from the ceiling is measured to be 4.5 metres and the platform was kept at a 1 metre distance from ground level which left a 3.5 metre distance from the platform to the top of the ceiling. It is worth mentioning that it is not important how far we keep the platform from the ground level, what is important is to keep that distance as a convention for the rest of our experiments.

We will also push the platform to force it to act as a pendulum. The system is asked to capture a reference patch in the first stage and keep that patch to apply the appearance comparison over the upcoming patches in the subsequent frames. However, as we explained earlier, the distance between the reference and the current images can be achieved by generating the Error Surface and finding the distance between the two frames and defining the lowest local minima in the surface.

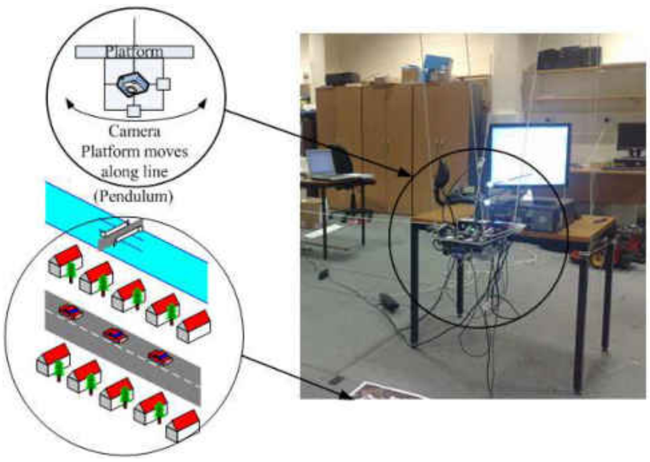


Figure 4.1 – Hanging platform over printed images

Passing the actual defined distance introduces further issues as the distance was defined in pixels, but the servos only understands the language of degrees. Therefore, we need to calibrate how many pixels make one degree or vice versa, in order to send the request to the platform.

The calibration process is required to enable the system to send the correct distance to the servos for it to move the camera by a correct distance. For our further experiments, we will use the lens with a 3.5mm angle view, in addition to using the 6mm lens to demonstrate how the larger angle's view affects the final outcome.

Figure 4.2 demonstrates our calibration scenario. Calibration begins by capturing the reference image, moving the servo(s) within certain distance then capture the second image to compare it against the stored referenced one and define the pixel displacement between the two images.

The next stage is to divide the servo's movement by the defined pixel displacement (D). The defined result of dividing the servo's movement by the defined pixel displacement is our defined calibrated number (C). After completion of the calibration loop, we will then use the calibrated number for our tracking system. The servo's movement is a straightforward process which is achieved after passing the calibrated values to the platform's servos. The values will move one or both servos to stabilise the camera. Figure 4.2 (right hand side image) demonstrates visually how the servos tries to re-stabilise the camera over the required location using the appearance comparison technique.

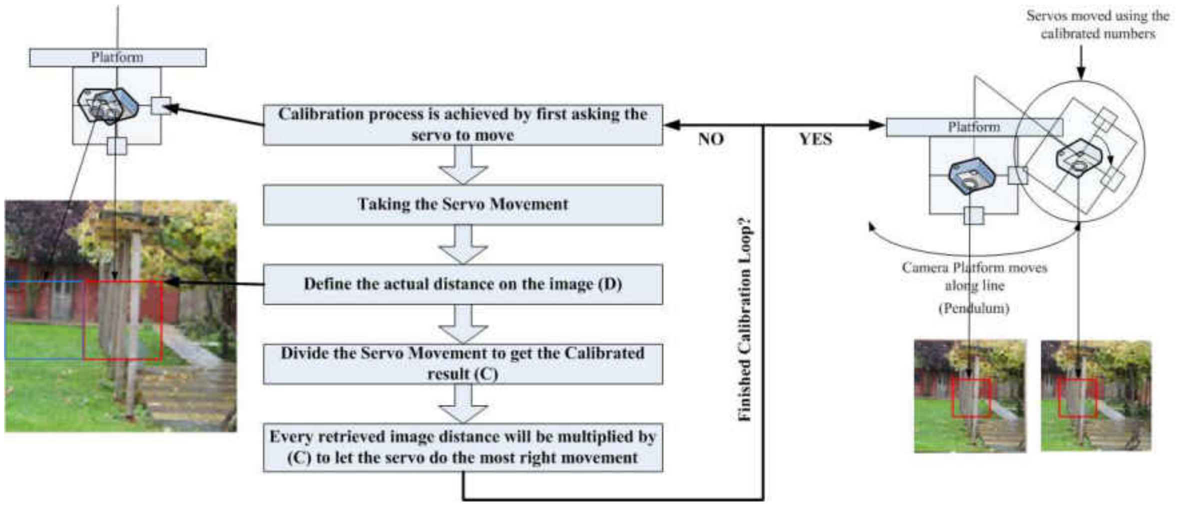

Platform
Calibration process is achieved by first asking the servo to move
Taking the Servo Movement
Define the actual distance on the image (D)
Divide the Servo Movement to get the Calibrated result (C)
Every retrieved image distance will be multiplied by (C) to let the servo do the most right movement
Finished Calibration Loop?
NO
YES
Servos moved using the calibrated numbers
Platform
Camera Platform moves along line (Pendulum)

Figure 4.2 – Calibration loop

Table 4.1 demonstrates the defined servo movements which we specified to go from -100 to +100, with a difference movement of 10. This process is repeated five times. Five pixel displacements for each movement were gained (e.g. D1...D5) and five calibrated numbers for each movement were also identified (e.g. C1...C5). In addition to this, as Table 4.1 also shows, the median for the calibrated numbers for all five trials were calculated and will be used for the next upcoming experiments. This calibration was made using the 3.5mm lens.

Table 4.1 – Servo movements for the calibration purpose

Servo Move	D1	C1	D2	C2	D3	C3	D4	C4	D5	C5
-100	-94	1.041667	-94	1.06383	-90	1.06383	-90	1.111111	-96	1.111111
-90	-84	1.046512	-86	1.071429	-80	1.046512	-86	1.125	-86	1.046512
-80	-74	1.025641	-76	1.081081	-76	1.052632	-76	1.052632	-78	1.052632
-70	-66	1.129032	-64	1.060606	-64	1.09375	-64	1.09375	-62	1.09375
-60	-56	1.111111	-54	1.071429	-54	1.111111	-56	1.111111	-54	1.071429
-50	-46	1	-50	1.086957	-50	1	-50	1	-50	1
-40	-38	1	-40	1.052632	-40	1	-40	1	-40	1
-30	-28	0.9375	-30	1.071429	-32	1	-30	0.9375	-32	1
-20	-24	0.909091	-22	0.833333	-22	0.909091	-22	0.909091	-22	0.909091
-10	-16	0.714286	-14	0.625	-12	0.714286	-14	0.833333	-14	0.714286
0	-4	0	-4	0	-4	0	-4	0	-4	0
10	6	2.5	6	1.666667	4	1.666667	6	2.5	4	1.666667
20	14	1.666667	14	1.428571	14	1.428571	14	1.428571	12	1.428571
30	22	1.666667	22	1.363636	18	1.363636	18	1.666667	18	1.666667
40	28	1.538462	30	1.428571	28	1.333333	26	1.428571	26	1.538462
50	36	1.470588	36	1.388889	36	1.388889	36	1.388889	34	1.388889
60	44	1.363636	46	1.363636	44	1.304348	44	1.363636	44	1.363636
70	52	1.346154	54	1.346154	52	1.296296	52	1.346154	52	1.346154
80	58	1.428571	62	1.37931	62	1.290323	58	1.290323	56	1.37931
90	66	1.363636	66	1.363636	66	1.363636	64	1.363636	66	1.40625
100	74	1.351351	74	1.351351	74	1.351351	74	1.351351	74	1.351351
Median (C1, C2, C3, C4, C5) = **1.11**										

For further set up procedures, we are aware of useable mechanical solutions within the Kite projects (i.e. Picavet). Mechanisms like the Picavet are well-known solutions to get the platform vertically stable. It is achieved by attaching the camera to the kite line and not the kite itself. The angle of the line to the kite is constantly changing. The camera cradle hangs beneath the Picavet cross from a bolt that is fastened through a hole at the centre of the cross. The Picavet cross provides a level platform for the camera cradle [2]. However, Picavet will keep the camera's head vertically downwards, whereas our solution tends to always keep the camera's head pointing towards the original location.

4.3 Patch Size Selection (PSS)

Looking at the Appearance Comparison chapter (Chapter 3), it was clear that the large patch has removed a lot of noise from the surface and made it much smoother and easier for such controllers demanding a gradient to navigate and reach the minimal. The penalty is the dramatic increase in the computational cost. However, the issue of computational cost was sorted by designing the random distribution patches, which required less pixel processing but covered a larger area. In addition to the computational cost of the large size patches, the ability to catch up with the movement speed is less possible with a large size image, as opposed to a smaller size one.

In order to select our right patch size, we would need to process this selection by first looking at which patch size is the best to be tracked. Figures 4.3, 4.4 and 4.5 are used to show the possible coverage of some patch sizes over the tracking process.

Best practice is to run experiment multi times and make sure that random variations are accounted for. However time consuming, nature of our working environment (consistent condition) and condition of our lighting resulted to have the decision to run single experiment.

Figure 4.3.a shows the entire image was captured by a large patch size and 4.3.b shows the shifting procedure towards the left where it is now impossible to get the entire reference patch matching within the current image's space. This makes the comparison process between the reference and the current patches incomplete, where a large original image's view was outside of the boundary. This means that only part of the two patches is correlating within the image's space and a faster movement may decrease the amount of similarities.

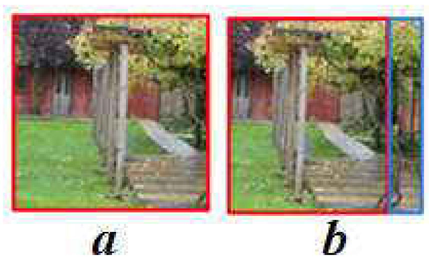


Figure 4.3 – Patch covering the whole image space

However, as Chapter 3 explained, the larger patches capture larger image's space which decreases the possibility of having more than one minimal point which would then reflect to have a smoother error surface with a clear single minimal distance. Yet, on the other hand, from the image processing point of view, covering larger image's space usually means processing a larger number of pixels unless some clever technical methods are used to get a great potential gradient with less pixel processing. Consequently, a balance between choosing the right patch size and the possible increase of pixel processing is mandatory.

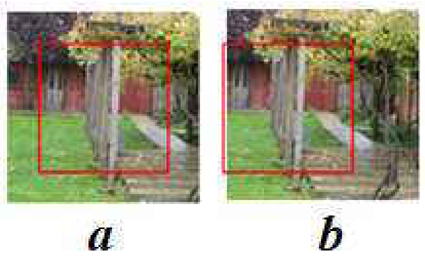


Figure 4.4 - Patch covering half of the image space

Figures 4.3, 4.4 and 4.5 demonstrate how the patch sizes of *300* × *300*, *150* × *150* and *100* × *100* were relocated back to the original location, after the camera was shifted. It demonstrates how the patch size of *100* × *100* was relocated with none of its borders outside of the margin. Therefore, we can take this as a basic argument for using the *100* × *100* patch size to handle the camera's shifting procedure better than the larger sizes (*300* × *300* and *150* × *150*). However, referring back to Chapter 3, we did use a patch size of *250* × *250* to generate a good error surface with a large area of potential gradient that the controller can use, but the question remains, which patch size is best for real time experiments?

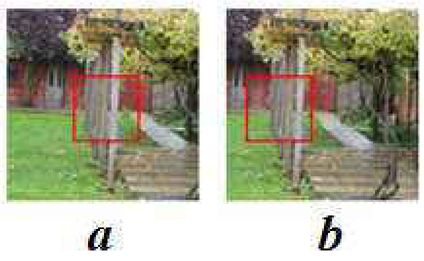
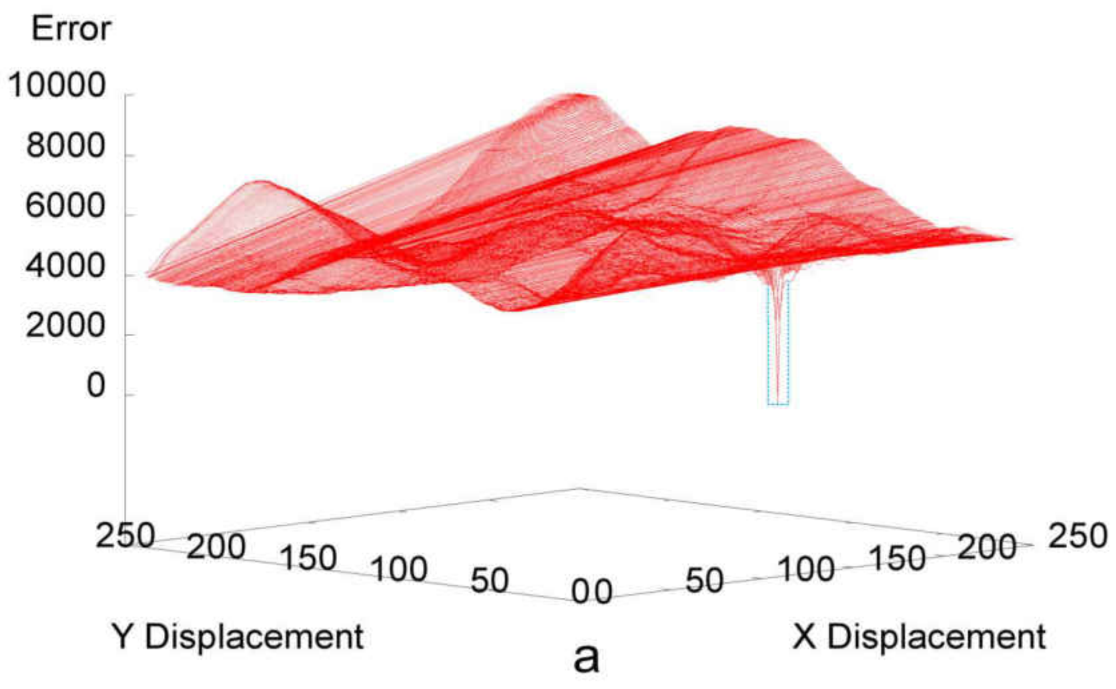


Figure 4.5 - Patch covering a third of the image space

Figure 4.6 shows the generated Error Surfaces using both *100* × *100* and *250* × *250* patches. The dotted rectangles in both 4.6.a and 4.6.b show the gradient descending ranges. Visually speaking, 4.6.b has a much larger dotted box than the one in 4.6.a. This means that the gradient descending range on the generated surface with the *250* × *250* patch has covered a larger image's space and therefore it has provided a better chance of finding the original target. However, we should be aware that these two surfaces were generated offline and we would need an online methodology to determine the trade-off between enlarging the patch size and handling the movement speed.



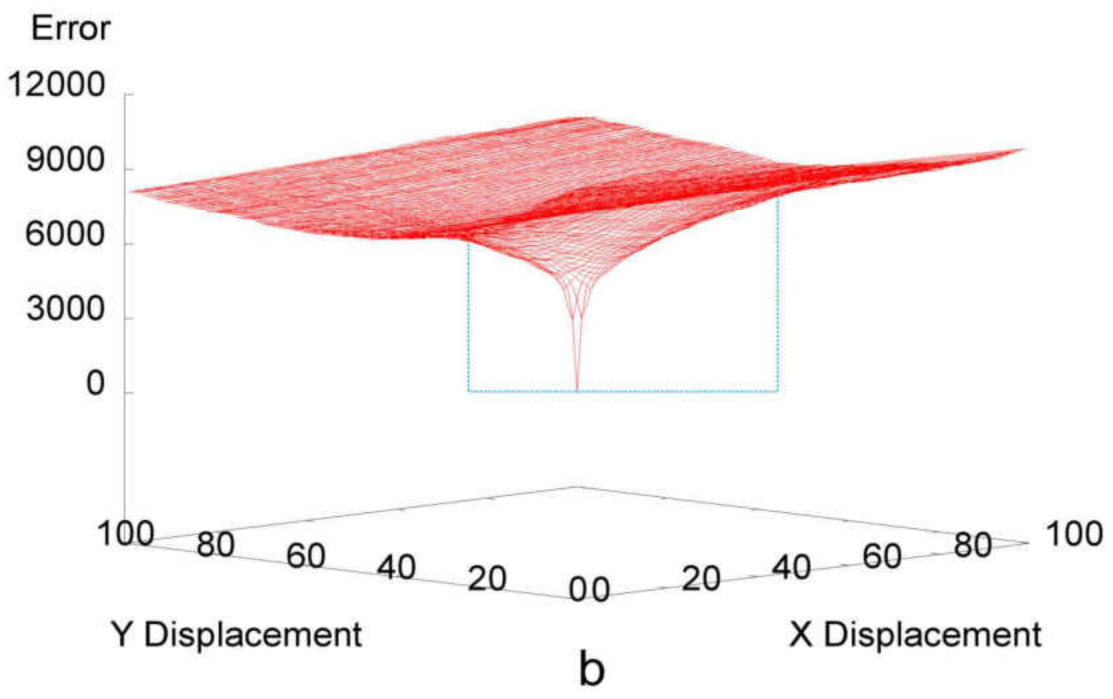


Figure 4.6 - Gradient comparison between two error surfaces generated from two different patch sizes. Two dotted rectangles demonstrate the range of the gradient on each Error Surface

To begin the online analysis, we first disable the platform's servos and allow the platform to move along the line for a short period of time. Meanwhile, the platform captures images continuously and calculates the pixel displacement between every two consecutive images.
Figure 4.7 shows two states where the platform is either pointing vertically downwards or pointing to the side. The two states differ from the swing's speed point of view, where the highest speed is when the platform is pointing fully downwards (i.e. central position) and gets to its lowest speed when it reaches the side. Hence, we are comparing every two consecutive images. We should expect the displacement to be higher when the platform is in the central position and lower when the platform reaches the sides. Figure 4.7 shows this as displacements are going from lowest to low to high and to highest, then vice versa.

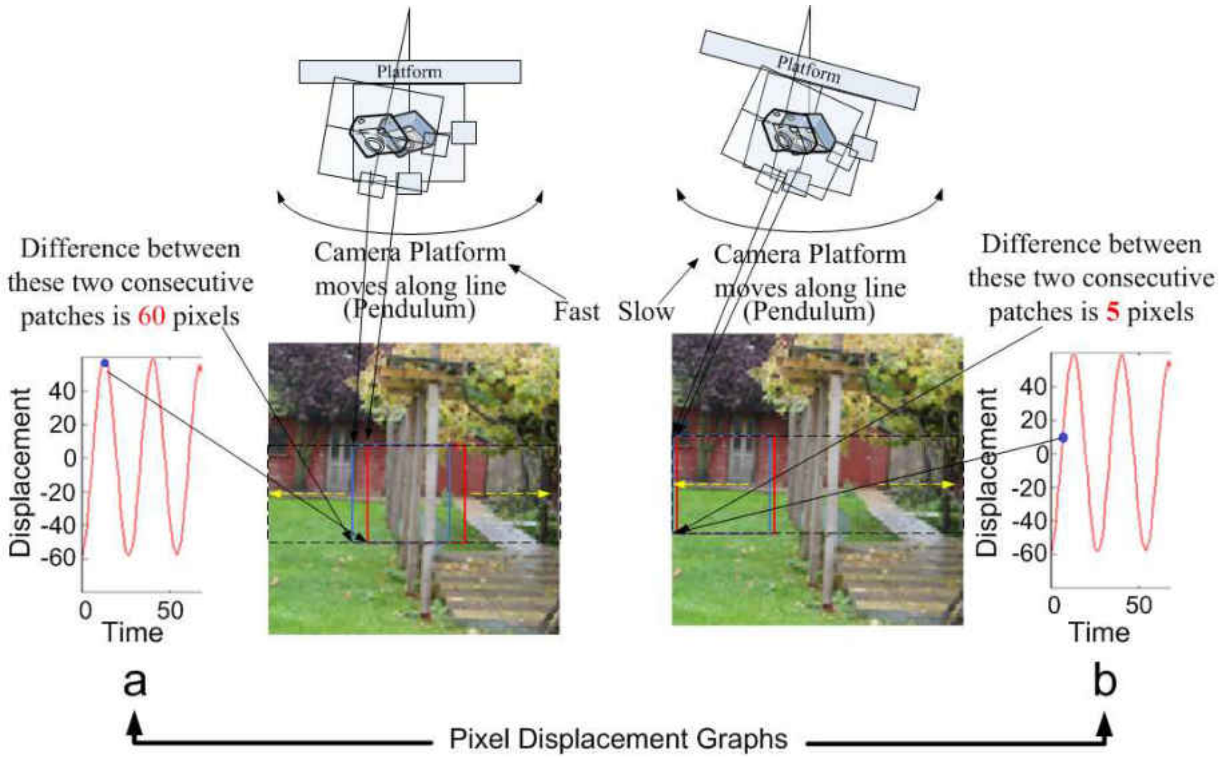

Platform
Platform
Difference between these two consecutive patches is 60 pixels
Camera Platform moves along line (Pendulum)
Fast
Slow
Camera Platform moves along line (Pendulum)
Difference between these two consecutive patches is 5 pixels
Displacement
40
20
0
-20
-40
-60
0
50
Time
a
Displacement
40
20
0
-20
-40
-60
0
50
Time
b
Pixel Displacement Graphs

Figure 4.7 – Pixel displacement graphs are made to represent the platform's possible position within the pendulum's cycle – E.g. Graph "a" showing the platform reached the central part of the pendulum where the pixel displacement between the two captured consecutive images is to be 60 pixels, but on the other hand, graph "b" shows how the pixel displacement between the two consecutive images is now only 5 pixels due to the high decrease in the swing's speed when the pendulum reaches one of its ends and the platform points towards the side.

In theory, if we plot the entire retrieved displacement numbers, the peaks should represent the time when the platform was in central position with the highest speed and the zero crossing represents the time that it reached the side with the lowest speed. Figure 4.7 shows an example of how the displacement is found to be 5 pixels on sides and raised to 60 pixels when the platform reached the central position. For future referencing, we will call the retrieved plot a "Pixel Displacement Graph" and the peaks "Direction Changes" (see Figure 4.8).

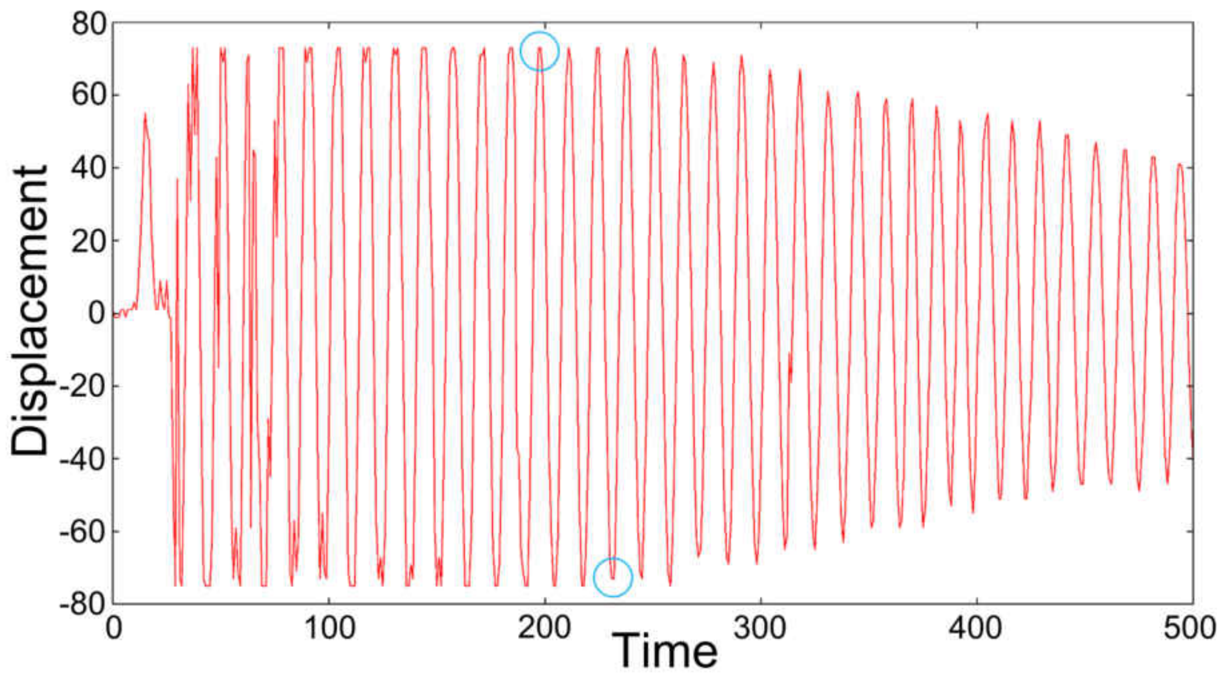


Figure 4.8 – Direction changes on the graph displacement graph

The Graph with the least found Direction Changes (DCs) is expected to be the smoothest, with accurate retrieved displacement numbers.

The hypothesis is that, the slice that generates plot with the fewer direction changes gives the better input stabilisation.

Previous figures (4.3, 4.4 and 4.5) have demonstrated the usability of having the patch size of (100×100) from the tracking point of view, but this may not be true in real world practice therefore, a new experiment aims to retrieve the pixel displacement graphs of three different patch sizes over three different images. The goal is to define which patch size {(50×50), (100×100), (150×150)} has introduced graphs with fewer DC values. This will indicate the usability of that patch size over real time tracking.

Figure 4.9 demonstrates the three images which will be printed and placed below the hanging platform (see Figure 4.1) to simulate the real time scenario.

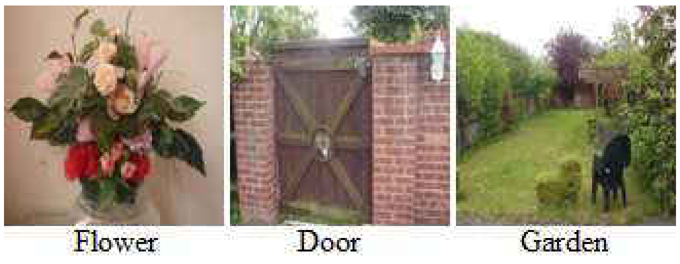


Figure 4.9 – Flower, Door and Garden images

Figure 4.10 presents the results of applying three different patch sizes over the Flower image. As mentioned earlier, the graph with the least Direction Changes is expected to represent better data quality, which in our case would be a better patch size to use. The first set of pixel displacement graphs are for the Flower image, as shown by Figure 4.10.

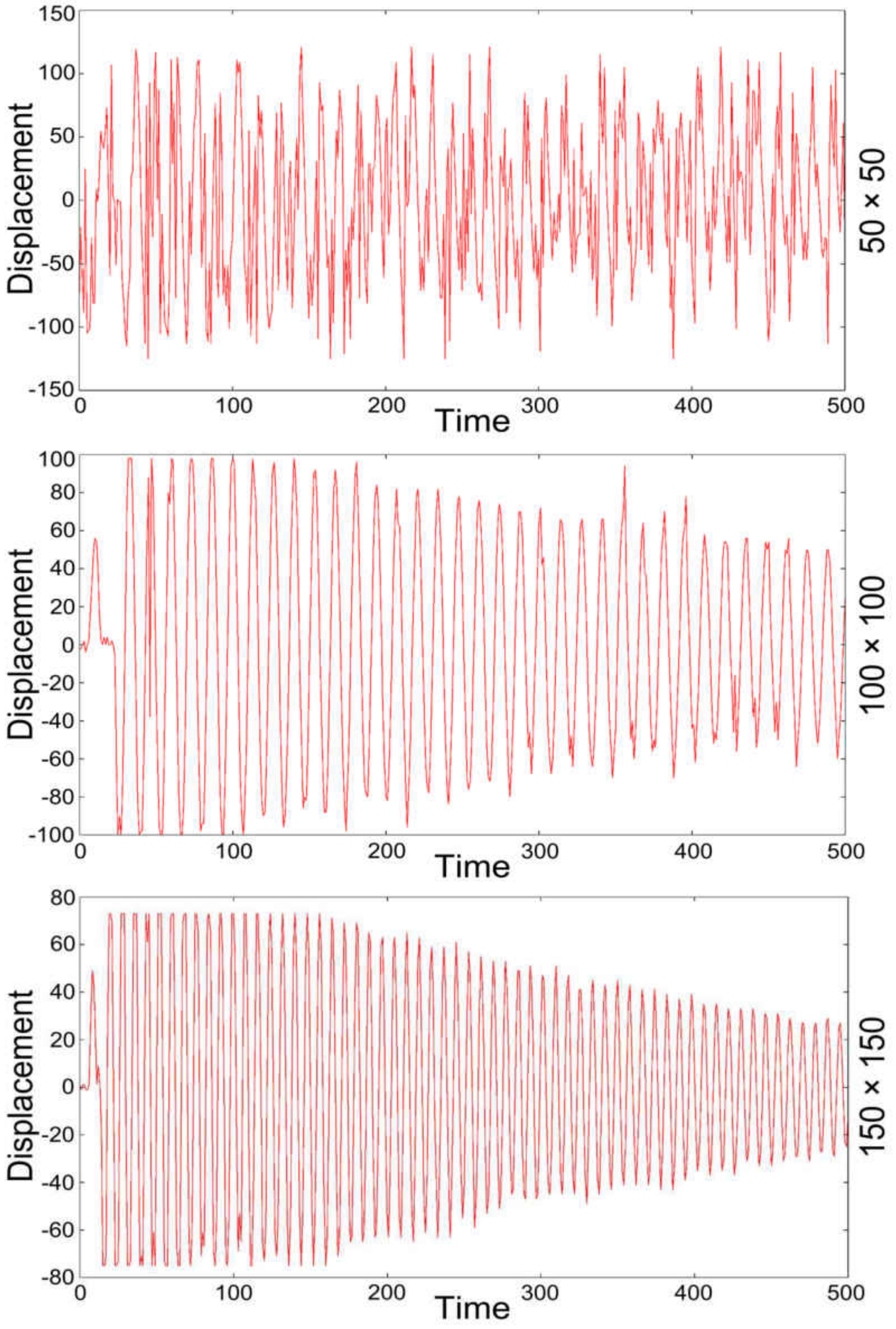


Figure 4.10 - Retrieved Pixel Displacement Graphs using three different patch sizes of *50 × 50*, *100 × 100* and *150 × 150* over the Flower image

As we are aiming to find the graph with the least number of Direction Changes (DCs), we will need to measure how many DCs are in each graph.

Table 4.2 shows the retrieved DC when three different patch sizes were used over the Flower image. It presents how the small patch (*50* × *50*) had the largest DC value (289 DCs) and that is mostly because the small patches can capture smaller areas, which tend to have a higher number of similarities across the image. However, getting more identical (or closed) values can redirect the navigation process to a different patch and draw the graph with many overshoots and noises (see Figure 4.10). On the other hand, when we had the largest patch size (*150* × *150*), we achieved much better data quality and reduced the DCs by a large number. Yet, referring back to Figures 4.3 and 4.4, it will become less possible to track the larger patch size in the subsequent frames. Therefore, running a smaller patch size (*100* × *100*) but one that is bigger than the first, (*50* × *50*) can be a good test to apply and obtain the required balance for the patch size.

Table 4.2 shows the results obtained from the patch size of (*100* × *100*) to be 123 DCs, which is less than both previous patches. This means that for this Flower image, the patch size of (*100* × *100*) was the best choice from all three sizes. Referring back, Figures 4.3, 4.4 and 4.5 will support the patch size selection and its track ability even further. However, due to having stable laboratory conditions, Table 4.2 demonstrates the results from a single run experiment.

Table 4.2 – Different patch sizes over flower image gaining different DC values

Flower	
Patch Size Strategy	**Direction Changes (DC)**
50	289
100	123
150	149

The second trial was done using the same patch size but over the Door image. We repeated the procedure and ascertained that the conditions were kept the same; for example, the surrounding illuminations would need to be similar as well as the distance that we pull or push the platform to simulate the pendulum. Figure 4.11 presents the pixel displacement graphs using the three patch sizes over the Door image.

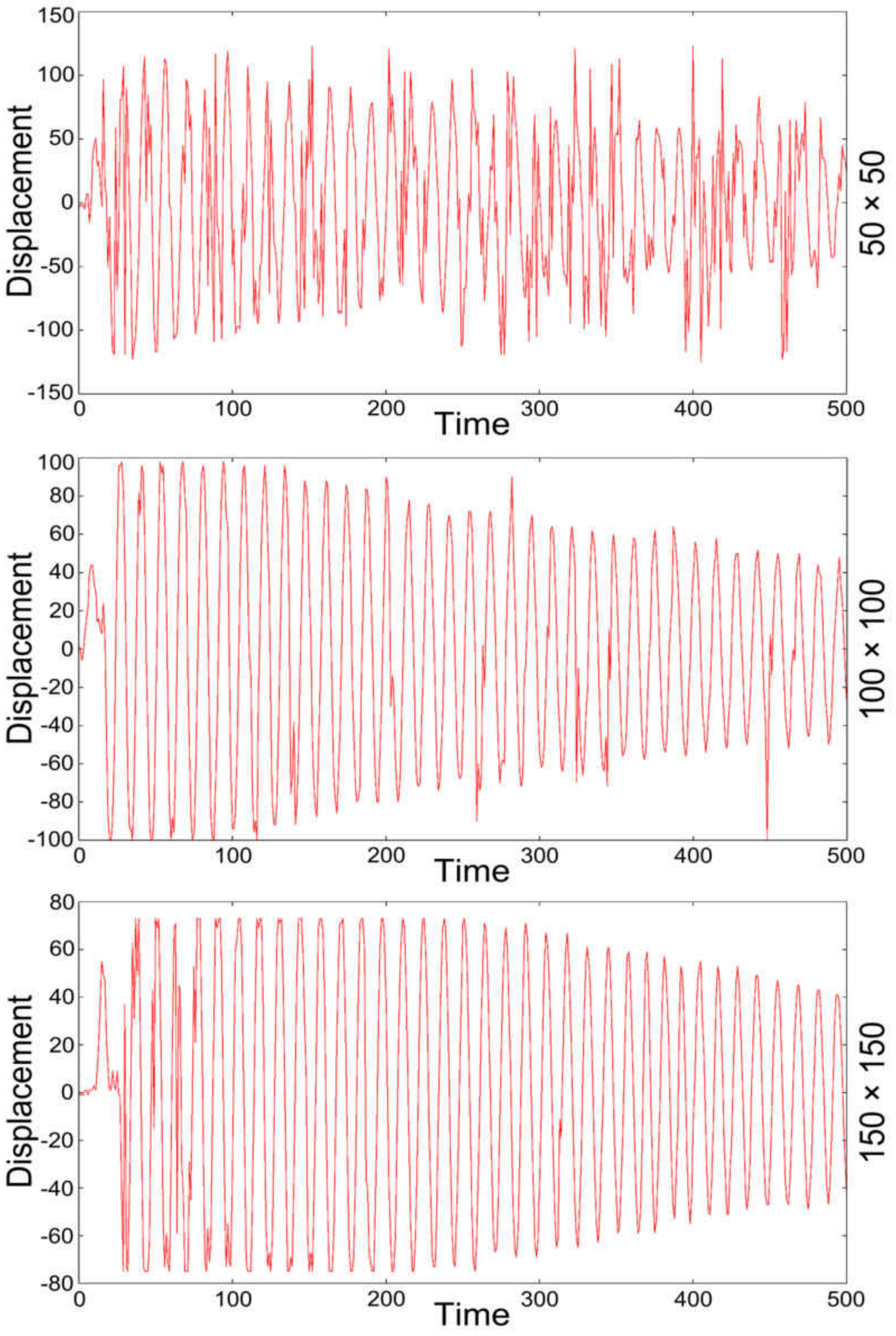


Figure 4.11 - Retrieved Pixel Displacement Graphs using three different patch sizes of *50 × 50*, *100 × 100* and *150 × 150* over the Door image

As Table 4.3 demonstrates, the ultimate outcome is quite similar to what we achieved with the Flower image. Also, similar to the scenario with Table 4.3, due to having a stable condition, the results were gained from a single applied experiment.

Table 4.3 - Different patch sizes over door image gaining different DC values

Door	
Patch Size Strategy	**Direction Changes (DCs)**
50	248
100	116
150	128

The final experiment was applied over the Garden image, and again, we used all three patch sizes and tried our best to keep the surrounding conditions similar as when the first and second experiments were applied. Figure 4.12 shows the three pixel displacement graphs using the three patch sizes.

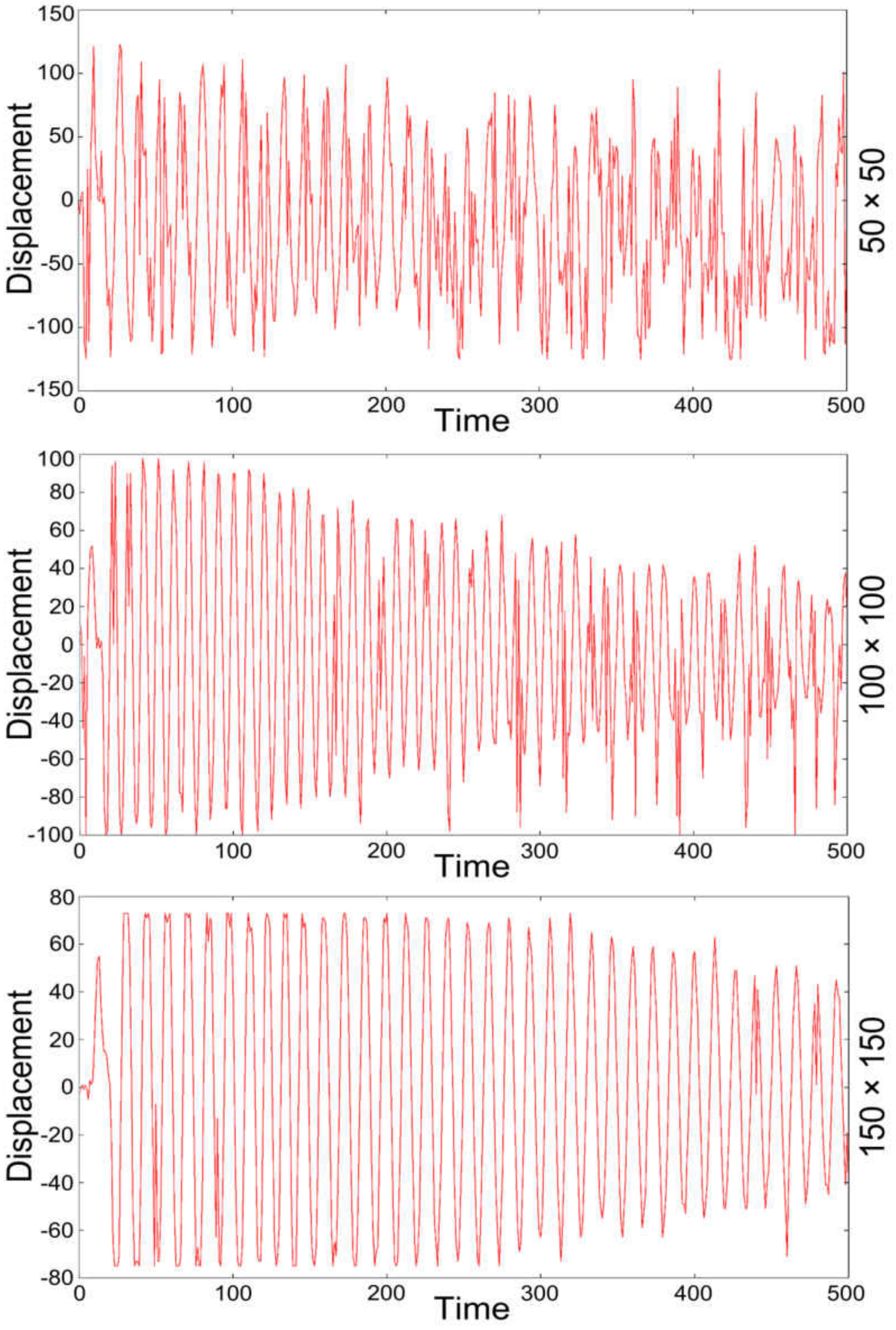


Figure 4.12 - Retrieved Pixel Displacement Graphs using three different patch sizes of *50 × 50*, *100 × 100* and *150 × 150* over the Garden image

Table 4.4 presents a slightly different outcome than the experiments over both the Flower and Door images. In this case, we still have the (*50* × *50*) patch size with the highest DCs (260 DCs) but the lowest DCs have now changed from being (*100* × *100*), as it was in previous cases, to a patch size of (*150* × *150*). The (*100* × *100*) patch size gained where the (*150* × *150*) had 121 DCs, making it the graph representing the best patch size for this scenario. As in previous cases, the represented numbers in Table 4.4 were gained from a single experiment.

Table 4.4 - Different patch sizes over garden image gaining different DC values

Garden	
Patch Size Strategy	**Direction Changes (DCs)**
50	260
100	183
150	121

When the platform is moving short distances at the end of the test, the patch is moving almost entirely on a uniform area of green in the image. Thus, poor matches are likely. The larger patch captures more variation. Consequently, Figure 4.13 demonstrates how the (*150* × *150*) patch could capture more variations than the (*100* × *100*) patch size. Part of the chair, door and the sky are examples of further variations the (*150* × *150*) patch included in its context.

However, after the theoretical discussion about Figures 4.3, 4.4 and 4.5, the real time experiments have shown the patch size of *100* × *100* to be a good size for handling the camera's frequent movement. Nevertheless, due to having quite well stabilised laboratory conditions, we then thought to have a single run for each patch size over each image. Therefore, each patch size was tested once on each of three different images, and the *100* × *100* patch size had quite a good outcome. In light of this decision the results that were obtained should be considered with respect to the small number of samples obtained.

Moreover, as we discussed earlier, more variations result in retrieving more accurate pixel displacements and higher accuracy means a smoother Graph with fewer Direction Changes (DCs). For this part of the experiment, the (*150* × *150*) patch generated fewer DCs which made it a better patch size than (*100* × *100*) and (*50* × *50*) patches (see Table 4.4).

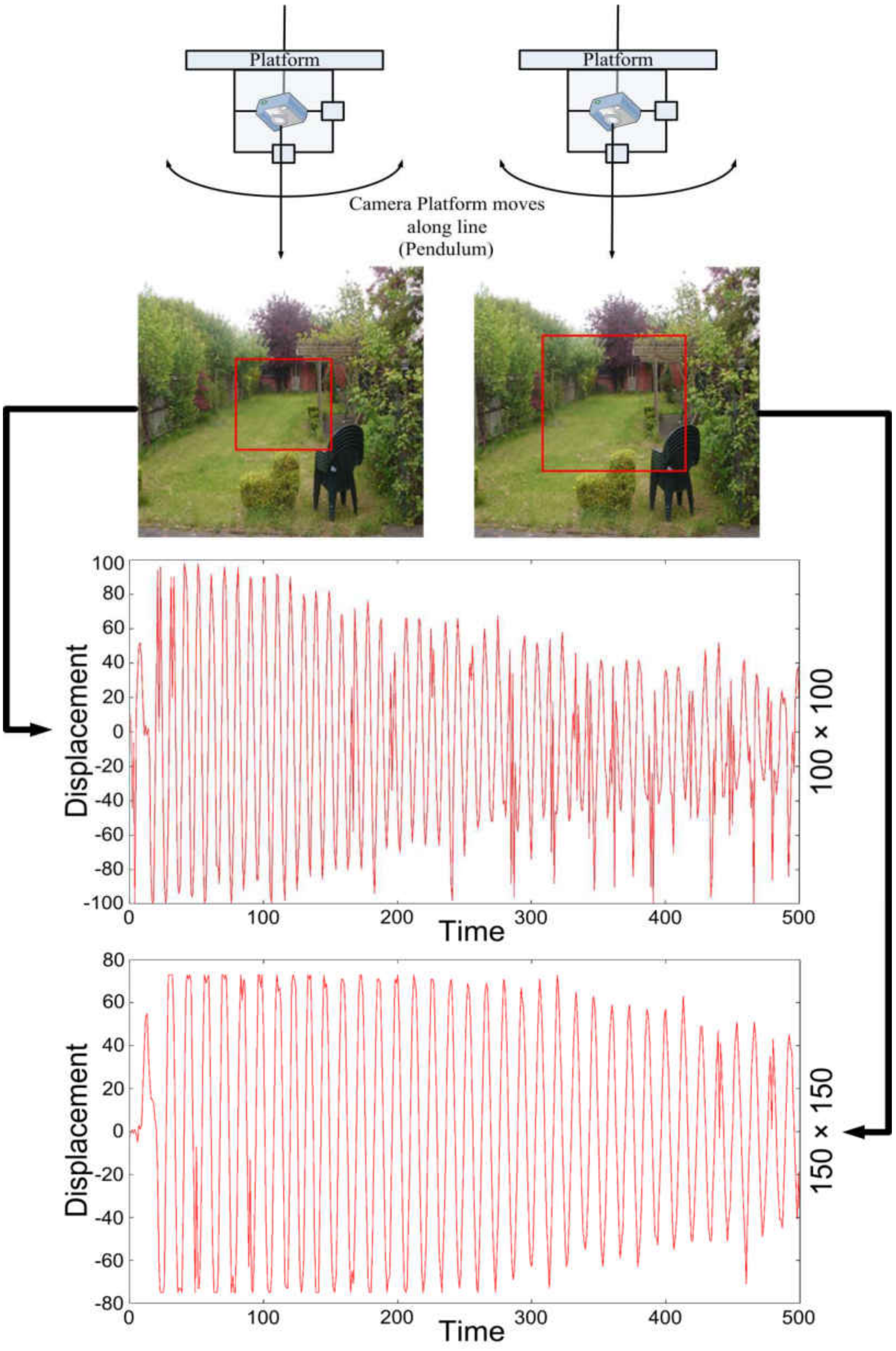


Figure 4.13 – Larger patch size reflects on generating smoother pixel displacement graph

With the results of the applied experiments on the three images using three different patches, the (*100* × *100*) patch size generated lower DC values in 2 out of 3 experiments. However, as we use the DC value as our core success criteria, the (*100* × *100*) patch size with the lowest DC values showed great potential of getting tracked in high speed motion, therefore, we can now declare the patch size of (*100* × *100*) to be an ideal patch size in all our further experiments.

4.4 Pixel Processing Reduction

An exhaustive search is expensive and we may still achieve a high quality outcome by iterating over the central X and Y axis only (see Figure 4.14). This will generate two useful curvy lines from the controller's point of view.

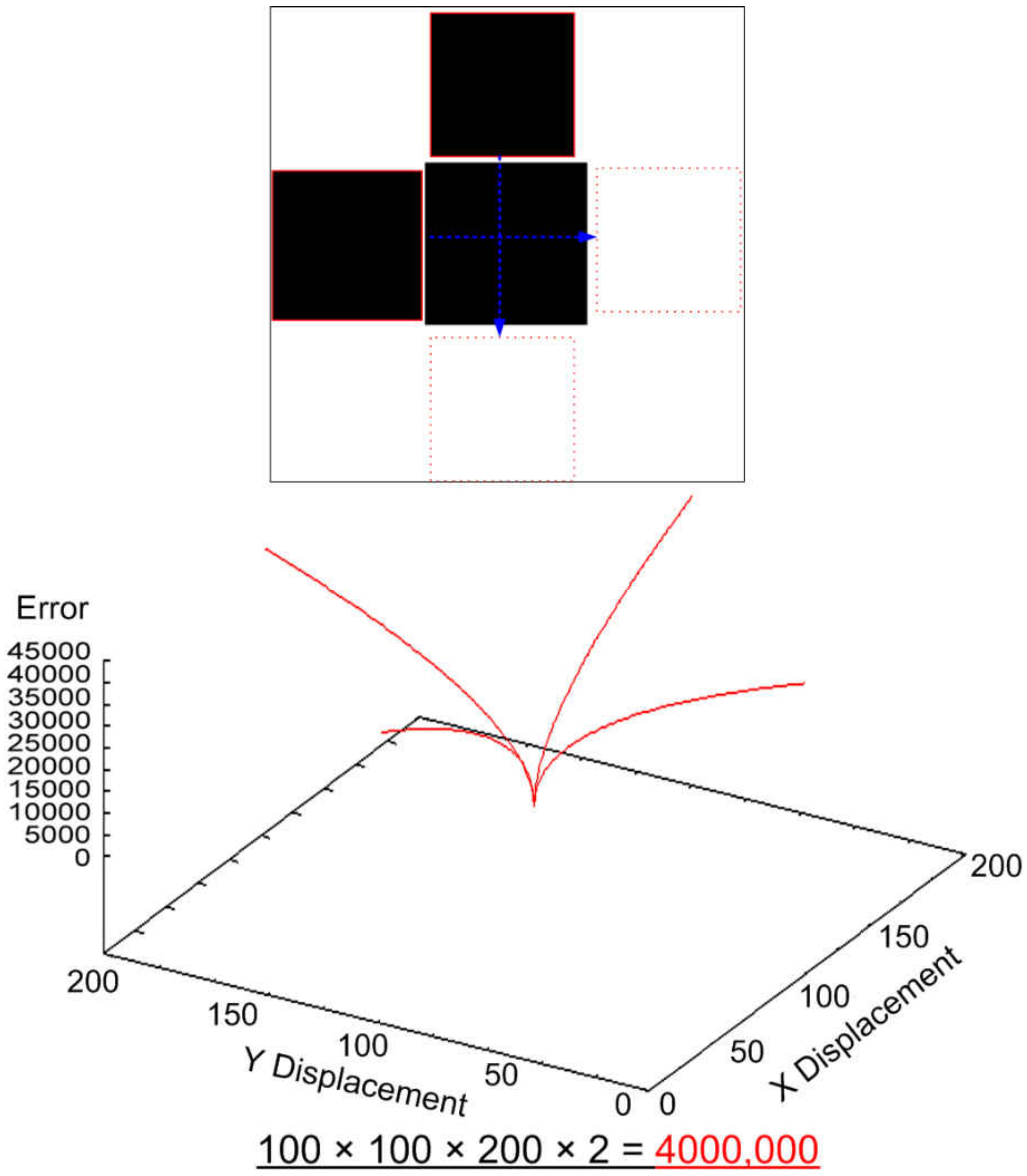


Figure 4.14 – Two curvy lines generated from patch iteration over the central X and Y axis only

In contrast to the X direction (pendulum direction), we require a different solution to resolve the non-stabilisation over the Y direction (see Figure 4.15). However, in contrast to a laboratory scenario, we can encounter much more non-stabilised conditions over the Y axis in the real world environment.

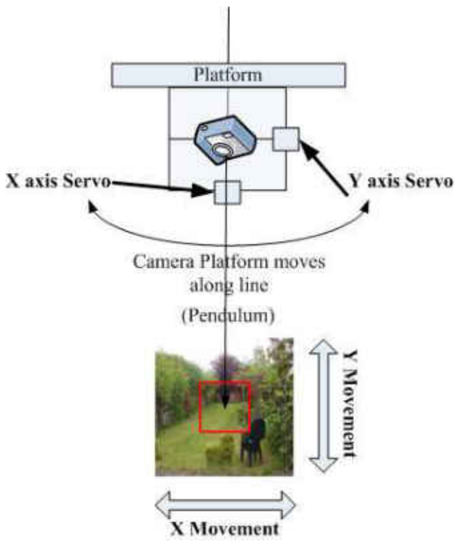


Figure 4.15 – Image's axes against the corresponding servos

Picavet solves some of the stabilisation issues for our Y axis but it certainly would not solve the entire concern related to the stabilisation on that axis. By implementing the Picavet on the Y axis, we are allowing the platform to resolve some of the stabilisation issues using a mechanical solution.

Having the pendulum on the X axis lets the patch to iterate through the X axis only. This generates a single curvy line that illustrates the local found minimum on the central X axis, which is still useable (see Figure 4.16).

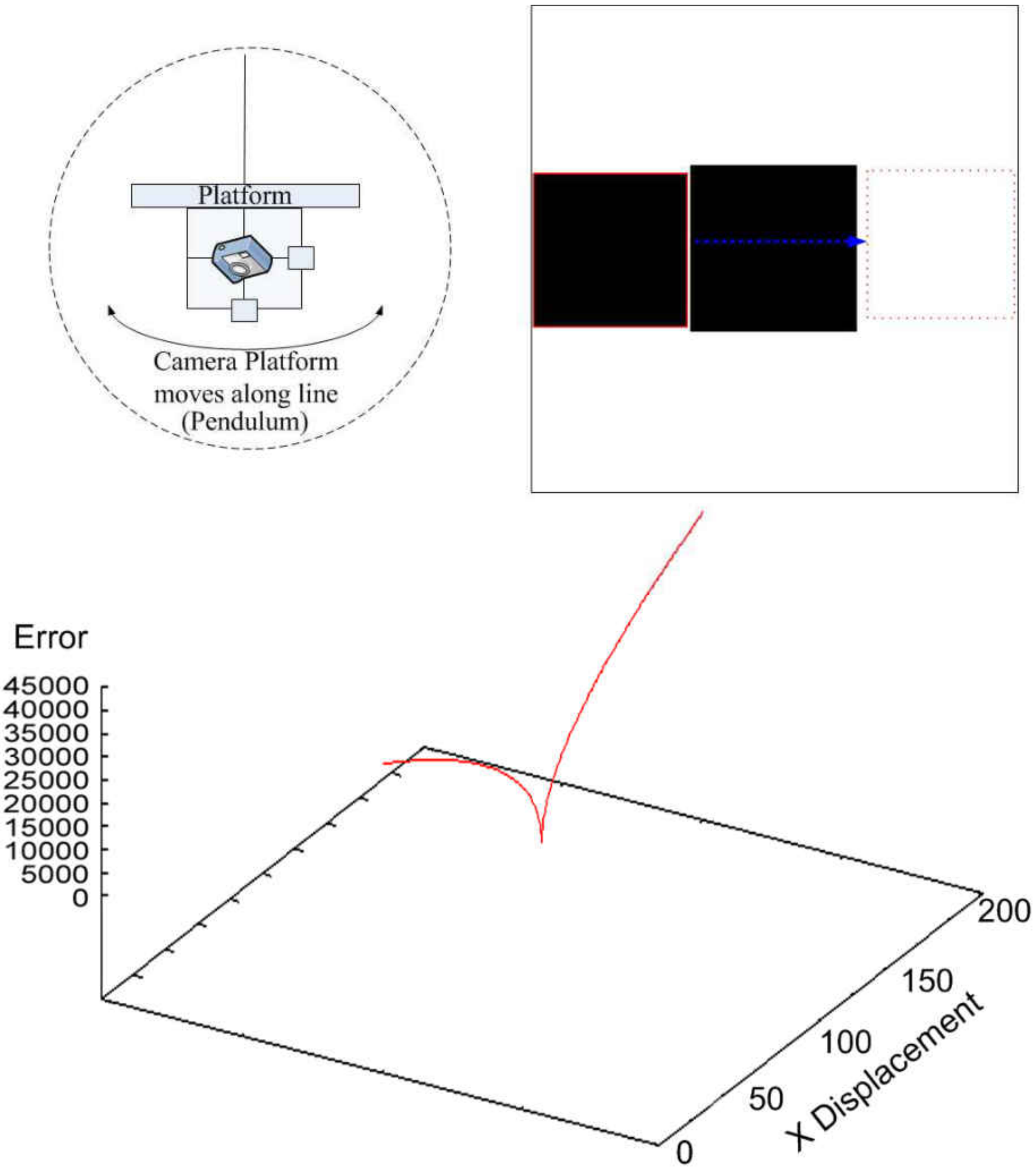


Figure 4.16 - Single curvy line generated from patch iteration over the central X only

Therefore, by taking into account that images are *300* × *300* pixels, we have reduced our search from 400,000,000 down to 2,000,000 pixel processing. This is equivalent to a 99.5% reduction and we will refer to it as “all” for future references (see Figure 4.17).

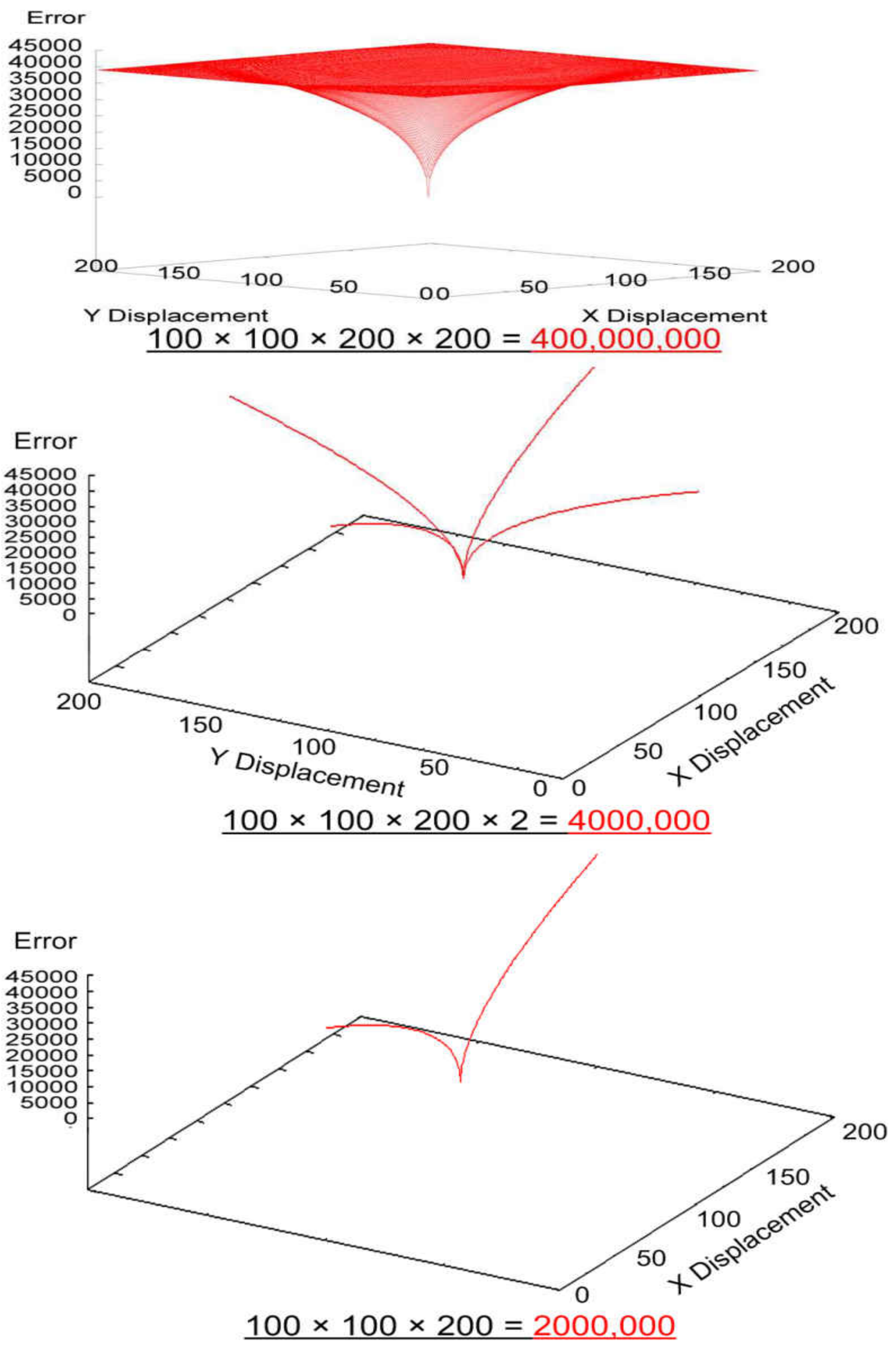


Figure 4.17 – Reduction from 400,000,000 down to 2,000,000 pixel processing

4.5 Hierarchical Search Strategies

Looking again at Figure 4.17 and specifically at the “all” strategy (where the entire central X axis was searched in order to find the minimal point) we now need to design methods for how to find the local point with less pixel search processing. We have designed hierarchal search strategies which supposedly substitute the exhaustive search, in the hope of achieving a similar performance. The strategies are titled “23”, “21”, “19”, “17” and “15”. Every strategy was named according to the number of steps it processes in every cycle. For instance, strategy “23” visits 23 locations and strategy “21” visit 21 locations and so on. To understand how they perform, we start describing the first Hierarchical search “23” which will also provide a clear view on how the others work. The methodology starts by locating the patch in the most central part of the X axis and then begins to shift the patch by the following displacements:

{-100, -80, -60, -40, -20, 0, 20, 40, 60, 80, 99}

Figure 4.18.b demonstrates this process on how the patch was shifted along the central X axis and also presents how this shifting generated an error curve to wholly exhibit the distance of the shifted patch from the central reference patch (see Figure 4.18.c). Arrows 2, 3 and 4 in Figure 4.18 show the relation between the patch shifting process and the generated error curve which would then be used to help us distribute the new shifting locations around the found minimal distance.

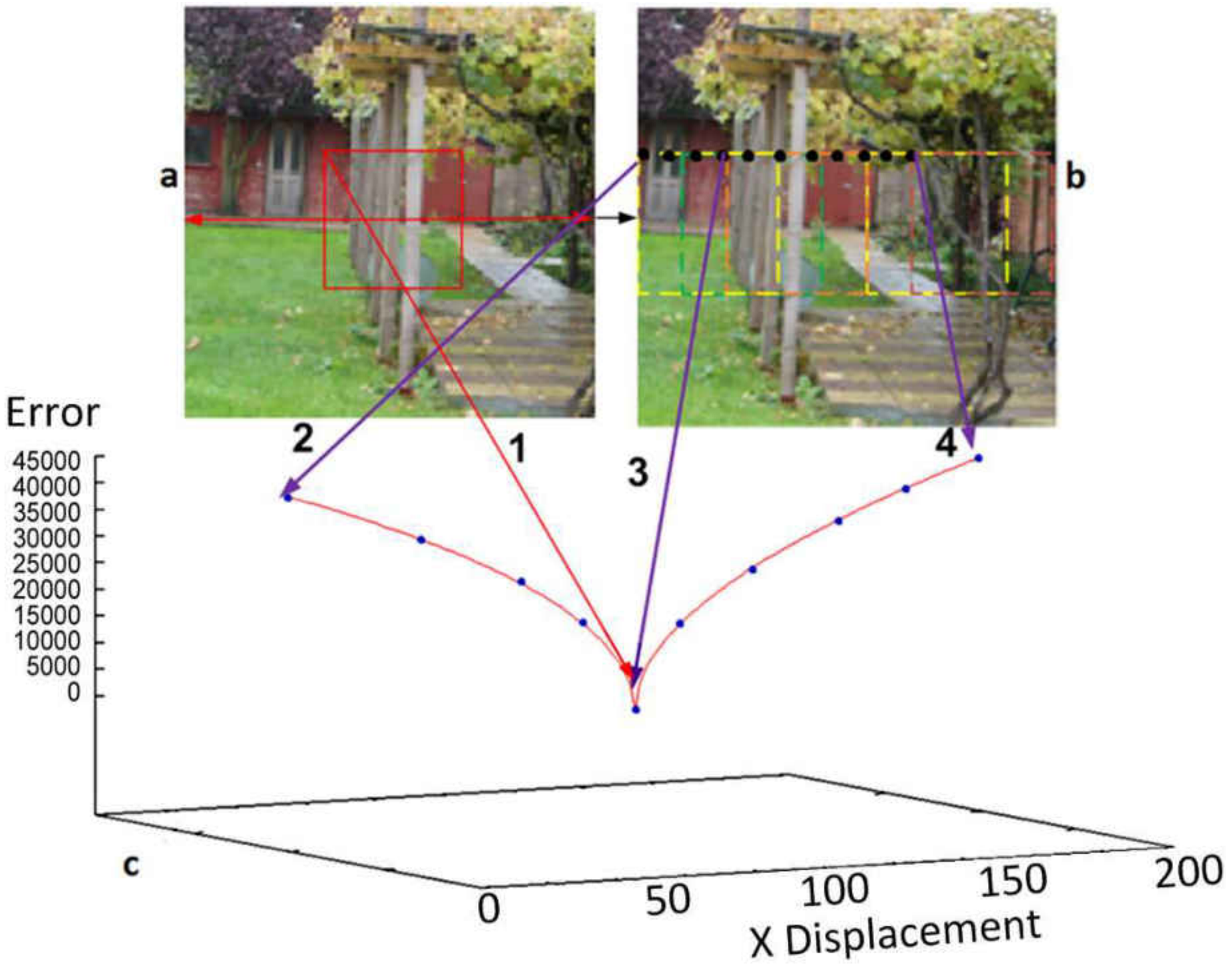


Figure 4.18 – First shifting iteration set

Figure 4.19.b now demonstrates the new patch shifting displacement {-15, -10, -5, 0, 5, 10, 15} around the found minimal point in Figure 4.18.

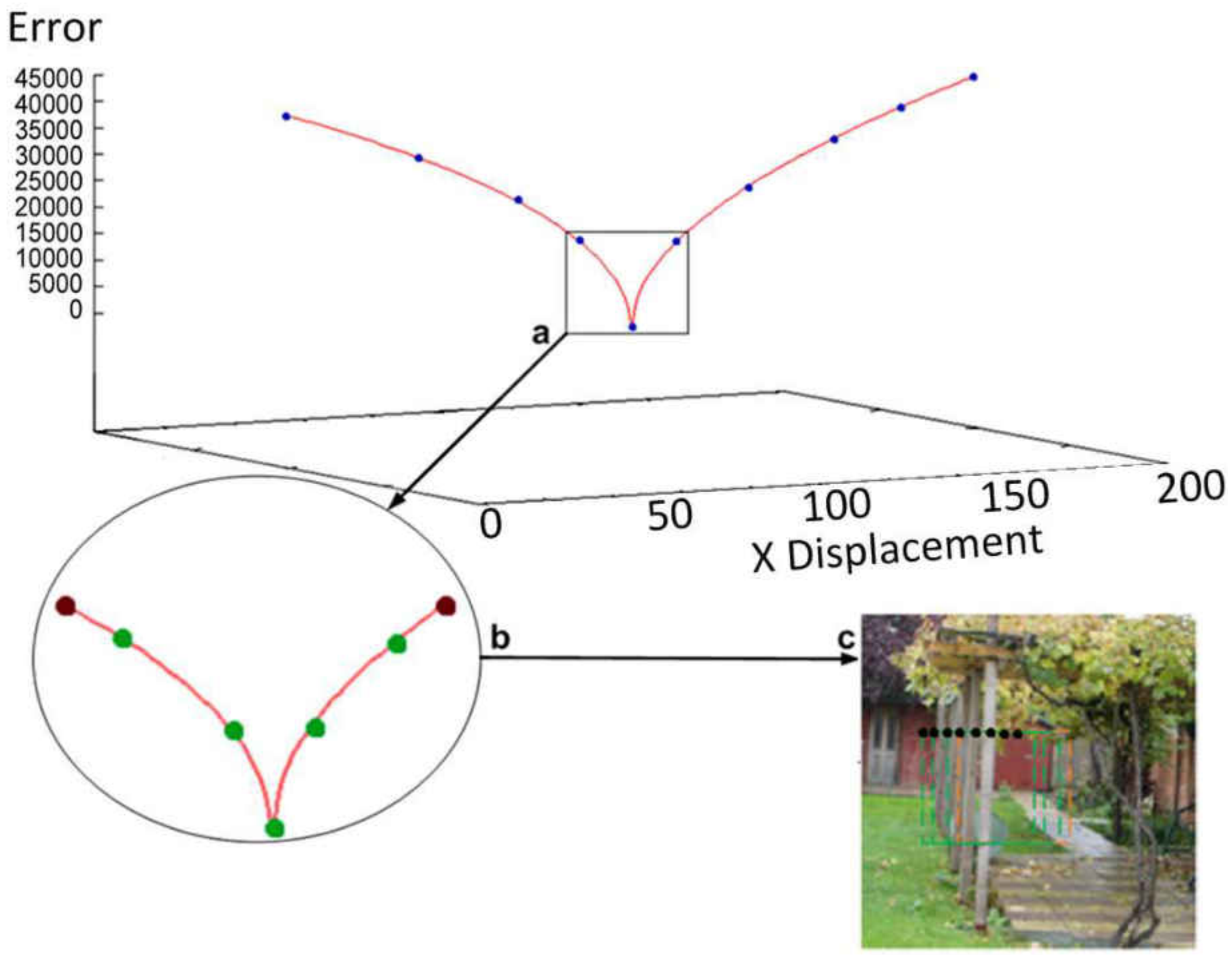


Figure 4.19 - Second shifting iteration set

The next step is to define the new shifting order from the new starting point {-4, -2, 0, 2, 4} (see Figure 4.20).

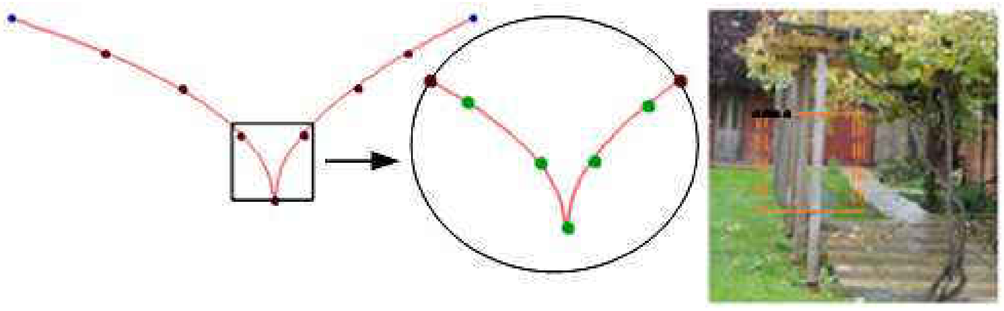


Figure 4.20 - Third shifting iteration set

In the last part, as Figure 4.21 shows, the minimal point will be defined amongst the ultimate shifting numbers (or positions) and will move the patch to the new specified point.

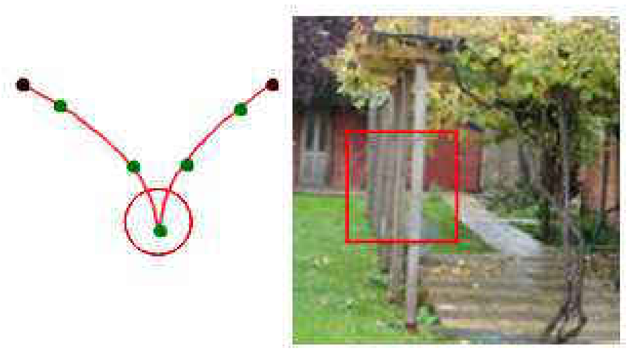


Figure 4.21 – Moving the patch to the best matching point

Eventually, we calculate the total distance that we moved the patch, from the beginning to the final point. The distance will be passed through to the servos to stabilise the camera and let it point to its original view.

We named the above hierarchical search as strategy “23” and that is due to the fact that we located the patch on 23 different locations over the entire axis, as a hierarchical structure to define the global minimal point. Table 4.5 shows the entire shifting steps for strategies “23”, “21”, “19”, “17” and “15”.

The way the rest of the strategies perform is exactly that same as how the “23” was designed, but they will be located over fewer and sometimes different locations.

Table 4.5 – Shifting steps for the hierarchical search strategies

23	-100	-80	-60	-40	-20	0	20	40	60	80	99	-15	-10	-5	0	5	10	15	-4	-2	0	2	4
21	X	-80	-60	-40	-20	0	20	40	60	80	X	-15	-10	-5	0	5	10	15	-4	-2	0	2	4
19	X	-80	-60	-40	-20	0	20	40	60	80	X	-15	-10	-5	0	5	10	15	-4	-2	X	2	X
17	X	-75	-50	-25	0	25	50	74	X	X	X	-15	-8	-1	X	6	X	13	-7	-4	-1	2	5
15	X	-75	-45	-15	X	15	45	74	X	X	X	-30	-20	-10	0	X	X	X	-6	-3	0	3	6

The hierarchical searching levels are indicated with various colours, where Table 4.6 shows the first hierarchical search level for the "23" strategy. When the patch is located on the most central position of the X axis, these numbers are then used to shift the patch along the X axis. It starts from -100 up to +99 (see Table 4.6). The number "0" is always the central position where no shifting process occurs.

Table 4.6

-100	-80	-60	-40	-20	0	20	40	60	80	99

We now have 11 results representing 11 Euclidean distances retrieved from applying 11 shifting numbers shown in Table 4.6. The least found distance will become our new central position (or "0" position). Table 4.7 is now showing new shifting numbers around the new "0" position. The 2nd hierarchical search level starts from -15 up to +15 (see Table 4.7).

Table 4.7

-15	-10	-5	0	5	10	15

Ultimately, Table 4.8 shows the final shifting numbers which are now around the new found least Euclidean distance.

Table 4.8

-4	-2	0	2	4

After clarifying how strategy "23" performs, we can now determine the way other strategies in Table 4.5 are executed.

However, since we have designed new searching techniques, the significant part is to estimate the saving made. Figure 4.22 demonstrates the number of pixels processed for the hierarchical and the "all" strategies.

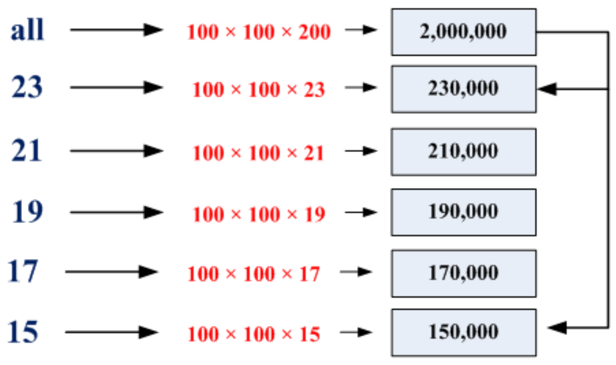


Figure 4.22 – Pixel processing for all six strategies

Figure 4.22 presents how the "all" strategy had 2,000,000 pixel processing. Also, the searching number was brought down to 230,000 pixels processing on strategy "23" and more savings were made over the rest of the strategies (i.e. Strategy "15" made 150,000 pixel processing).

In order to reduce the computational overhead associated with our current patches, the search was made over a single colour channel but some down sampling is also required. Once again, the use of complex techniques to combine pixels etc. is not appropriate for reasons of computational overhead. The use of a random (but fixed) distribution of pixels within the patch (see Chapter 3) rather than all the pixels was selected. This process of random distribution is applied to both the reference and current patches, which will reduce the number of pixels being processed enormously. Figure 4.23 shows an example distribution which captures 10 percent of the pixels for the comparison process, making a massive reduction over the pixel processing. For example, the "all" was decreased from 2,000,000 pixels processed down to 200,000 and "23" went down to 23,000 from 230,000 pixels being processed previously.

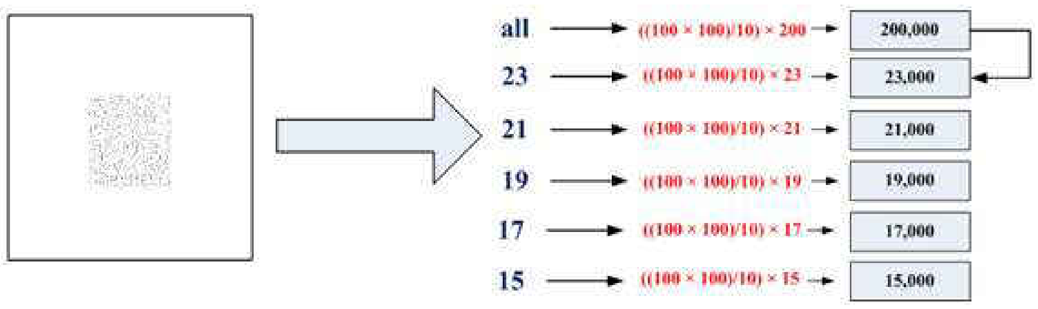


Figure 4.23 – Random distribution will process 10% of the entire pixels

The implemented hierarchical searches are slightly different to many existing methods like Gradient decent, Newton's Method and many similar minimisation methods which were designed to look for the local minima. Our hierarchical searches were designed to look for the global minimal point. Figure 4.24 is a good example of such an error curve which does have many minima. Methods like the gradient descent are fast enough to always be within the required local minimal but any fast manoeuvre can lead to shifting the localisation or the tracking process to a totally different direction. For example, if we use the error curve that is shown below for our tracking process, the gradient descent and such other similar methods can be shifted to any of the shown local minimal points (green circles). However, the hierarchical searches were designed to act like an exhaustive searching process but with much less pixel processing and a similar outcome.

Choosing the right patch size, using the SSP technique for the allocation process and employing the hierarchical searches can address the problems to resolve the stabilisation issues of using visual analysis.

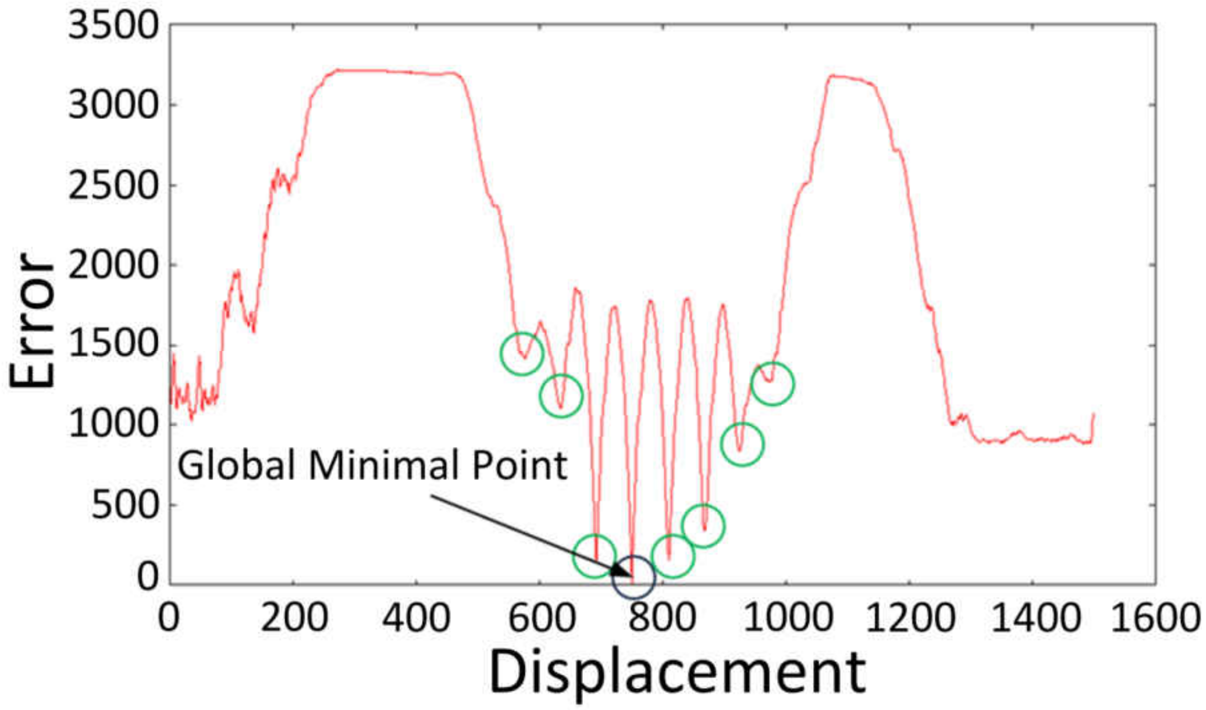


Figure 4.24 – Similar regions reflect on having more local minima

Figure 4.25 shows the five images which used to be located below the platform to allow the platform to stabilise over.

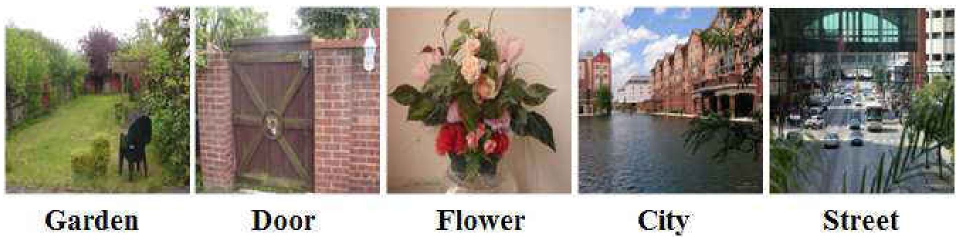


Figure 4.25 – Garden, Door, Flower, City and Street images to be printed and placed below the hanging platform

As Figure 4.25 demonstrates, the stabilisation results of strategies “all”, “23”, “21”, “19”, “17” and “15” are a, b, c, d, e and f using box and whisker plotting. Five runs were applied for each strategy then the results were compared to see if the designed strategies have a similar performance over each of the five images (see Figure 4.25). The comparison process has used the statistical P value. To retrieve the P value, we have used the Kruskal-Wallis [135] technique which is a non-parametric stat supplied by the MINITAB [136] software. If the gained P value is over 0.05, it means the correlation amongst our datasets is high enough and there is no difference amongst the performance of all 6 strategies over the used images. As a consequence of our comparison outcome, Figure 4.26 shows the P value on the Flower image failed to exceed the 0.05 margin. So, this means that, the hypothesis set out at the beginning of the chapter is not practical for all type of images. Many of the experiments seem to show good performance of the algorithm, but in cases where the image is either relatively featureless, or contains repeating patterns the algorithm is more likely to fail to maintain good stabilisation. The existence of very bland image regions in images such as the Flower image or repeating features such as the brickwork in the Door image are natural features of the world, and it seems inevitable that some images will not be effectively tackled using this technique alone.

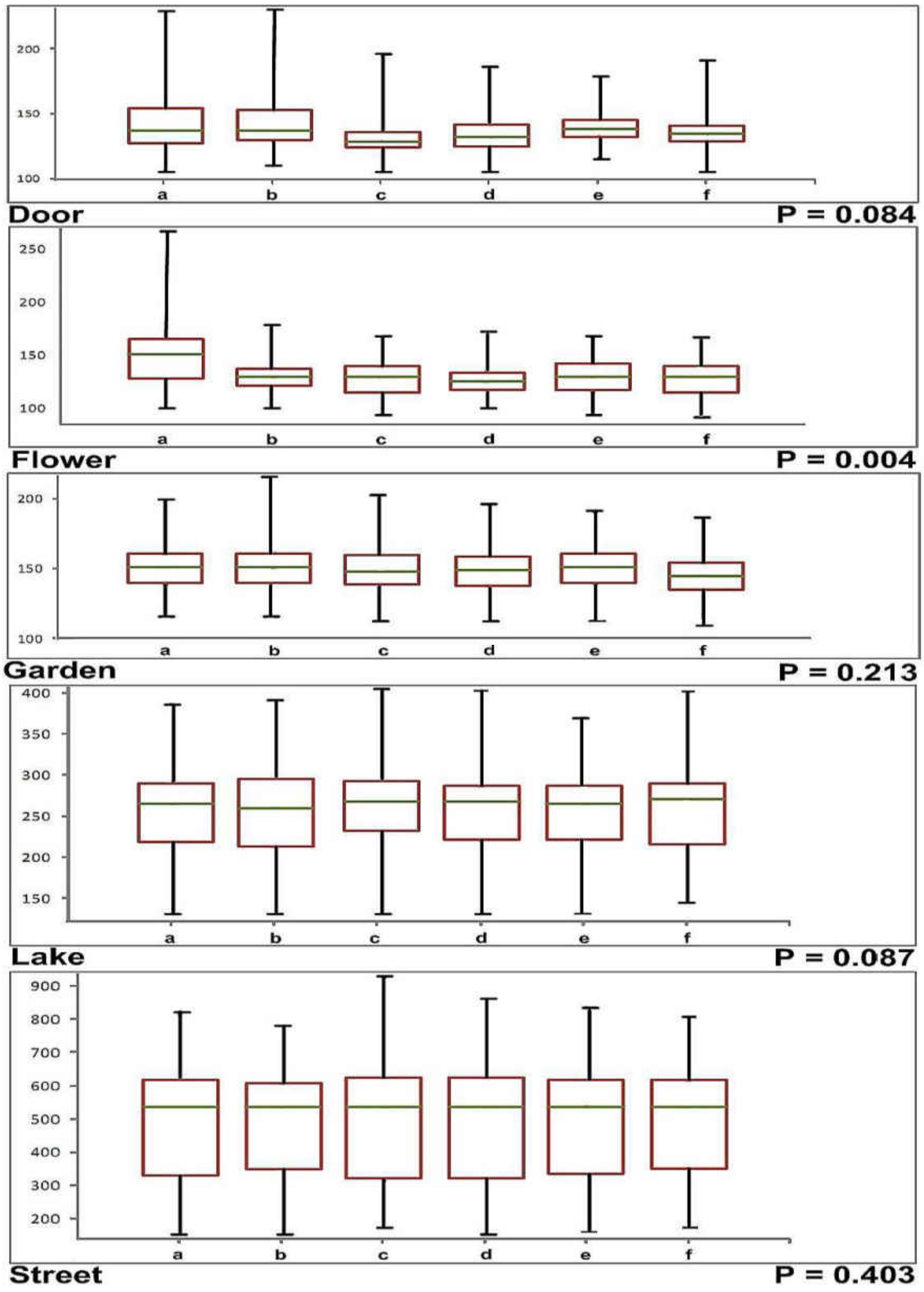


Figure 4.26 – Using box and whisker plotting to present the gained data

4.6 Slice Selection Process (SSP)

After describing how the process of Error Surfaces (or Error Curves) generation is achieved and how to use them to navigate over the target, we now need to take one step backwards and think of how to select our reference patch. The reference patch is the most essential part, as unlike the continuously captured images, the reference image will be used during the entire process to be compared against the upcoming frames. Therefore, it is highly important to determine how to choose the reference and is essential to verify if it is best to use the capture patch from the central part.

Figure 4.27 demonstrates an image where the black box is now located on the top central part rather than the most central, as in previous examples. If we now capture two patches, one to be from the most central (as in previous examples) and the other to be from the centre of the top, the generated error curve from the top layer (or slice) is more descending towards the minimal point. This means the controller has more opportunity to define the target when using the error curve with a smoother curve descending towards the minimal found error. The problem that we encountered with the central layer was because when we captured the central patch from the middle, the comparison process did not identify any difference between the captured reference patch and any other captured patch over the same axis. Yet, the scenario is different when the captured reference is applied over the most central location of the top layer. Here, the comparison process will find a large difference between the reference and any captured patch throughout the same axis. As a consequence of this, Figure 4.27 can present quite a high difference between the two generated error curves when the comparison ran over both layers, where the difference is descending and is visually quite well observable.

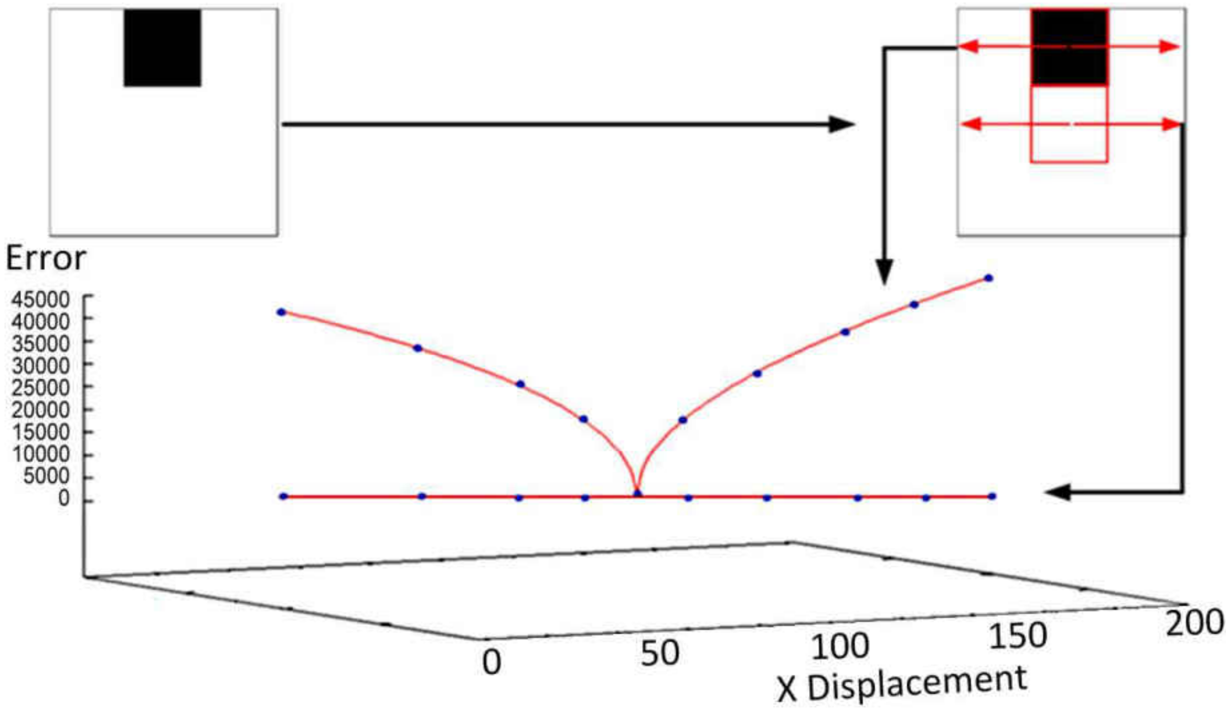


Figure 4.27 – Patch position reflects over the error curve shape

Knowing where to locate the reference patch is the foundation of the entire route, which provides the base of either having a strong or weak stabilisation. The requirement of knowing where best to locate the reference is a major prerequisite process.

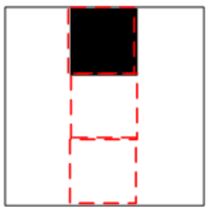


Figure 4.28 – Choosing where to locate the patch is an important step in the initialisation procedure

As for PSS, we start by letting the platform move along the line and keeping the servos disabled. The distance between the consecutive images is calculated while the platform keeps swinging but no physical stabilisation action is processed. The platform will move along the line for a short period of time while it captures images continuously. The image is divided into multiple horizontal sections. We call each of these sections a “Slice”. A Slice is a rectangular piece taken from the image view, where the height is only a small part of the entire height but the width is the actual image's width. Figure 4.29 demonstrates how the image is divided into multiple slices and the horizontal piece below the actual image is an example of what a single slice looks like.

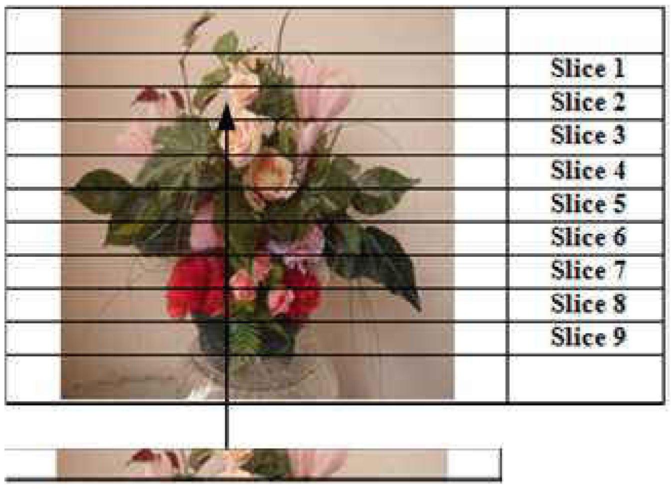


Figure 4.29 – Image divided into 9 slices

As with PSS, where the appearance comparison was processed over the central slice, we now apply the same process over the 9 slice divisions. Hence, we apply the process over 9 slices and expect 9 Pixel Displacement Graphs (see Figure 4.29) to be generated. The one with the lowest number of Direction Changes is expected to be the best slice for the platform to select the reference patch from. Figure 4.30 presents the SSP steps, which also indicates the need to set a deadline for the SSP method and terminate it.

If the graph has fewer Direction Changes, it means the colour variations within that slice are clearer to the platform's camera and the appearance comparison between the patches generates better Error Surfaces (or Error Curves). Ultimately, this means the patches within that slice are more useful.

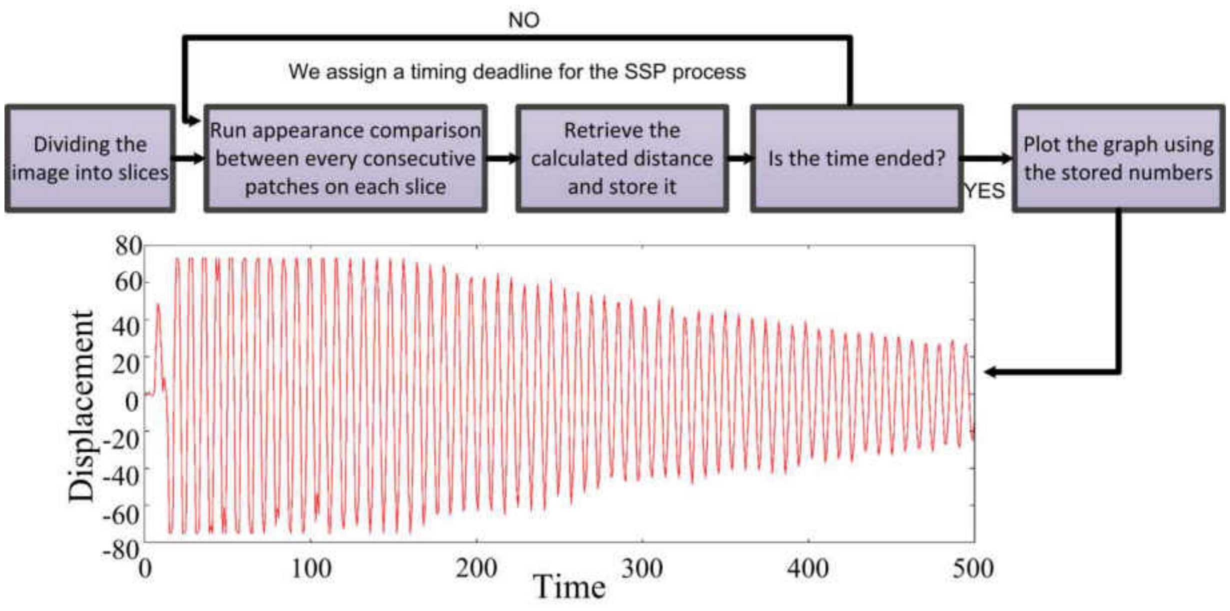

NO
We assign a timing deadline for the SSP process
Dividing the image into slices
Run appearance comparison between every consecutive patches on each slice
Retrieve the calculated distance and store it
Is the time ended?
YES
Plot the graph using the stored numbers
Displacement
Time
80
60
40
20
0
-20
-40
-60
-80
0
100
200
300
400
500

Figure 4.30 – SSP steps

Figure 4.31 shows the best and worst slices from the applied SSP test over the Flower image. As the number of Direction Changes helps to know which slice is best to stabilise on, the example in this Figure shows slice 6 to have the highest number of Direction Changes with 309, and slice 4 having the lowest with 87 DC.

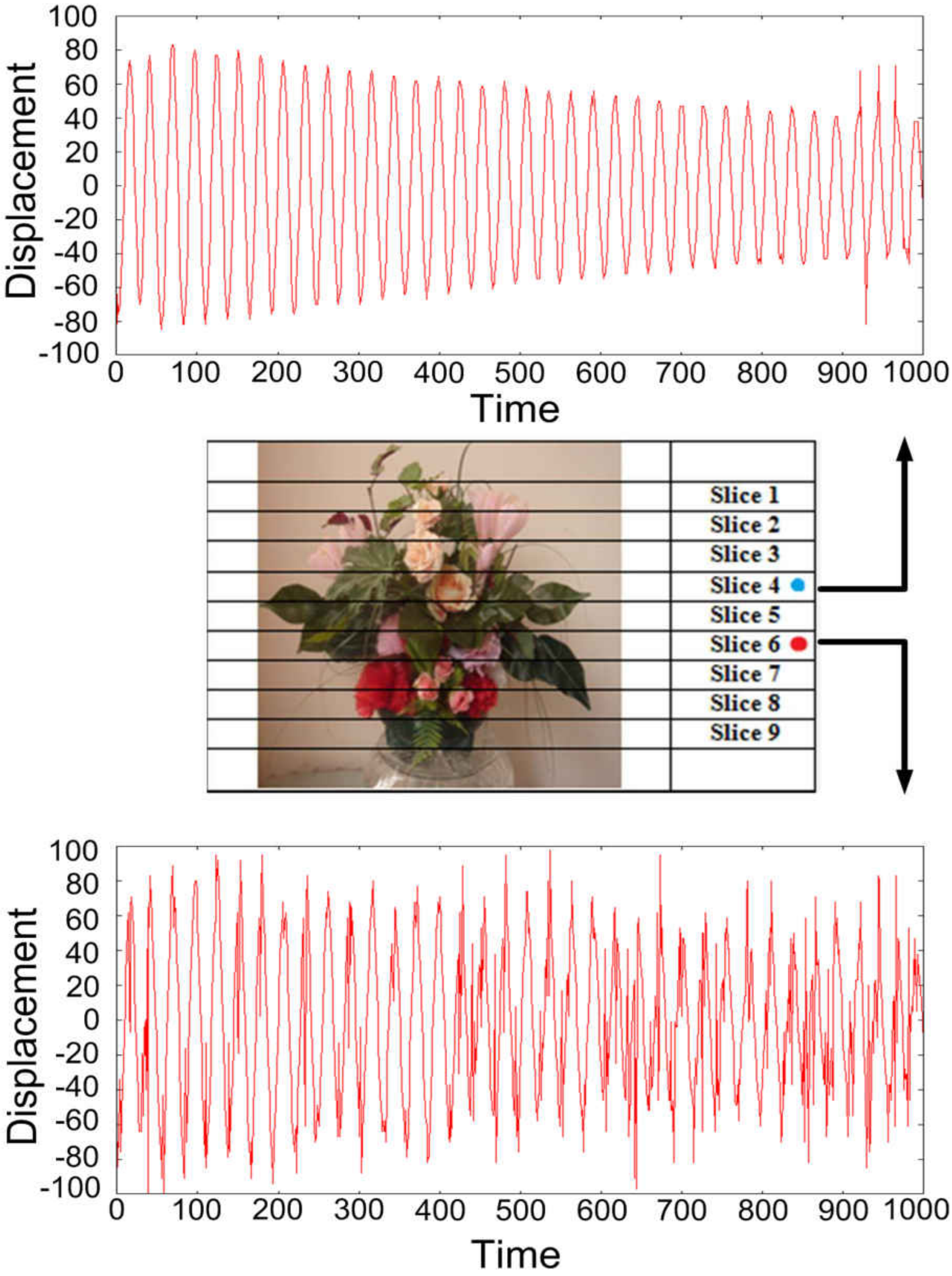


Figure 4.31 – Example of two pixel displacement graphs over two different slices

The next stage is to enable the servos and locate the reference patch over the best found slice (Slice 4). Figure 4.32 demonstrates how the reference patch was located over the best and worst slices. The same Figure presents the P value as 0.001, which is an indication of the difference between the two data sets (a & b). Dataset "a" refers to the outcome of the stabilisation technique over the best slice (slice 4) and "b" refers to the results of the applied stabilisation over the worst slice (slice 6). To determine which result shows a better outcome, we look at the dataset with smaller values. As the practice was the stabilisation procedure, the results are the distance between the reference and the current patch. Therefore, by looking at the presented analysis in this Figure, we can see that the dataset "a" was shifted more to the left, indicating less errors. This signifies that the stabilisation technique over slice 4 performed better than the applied technique over slice 6.

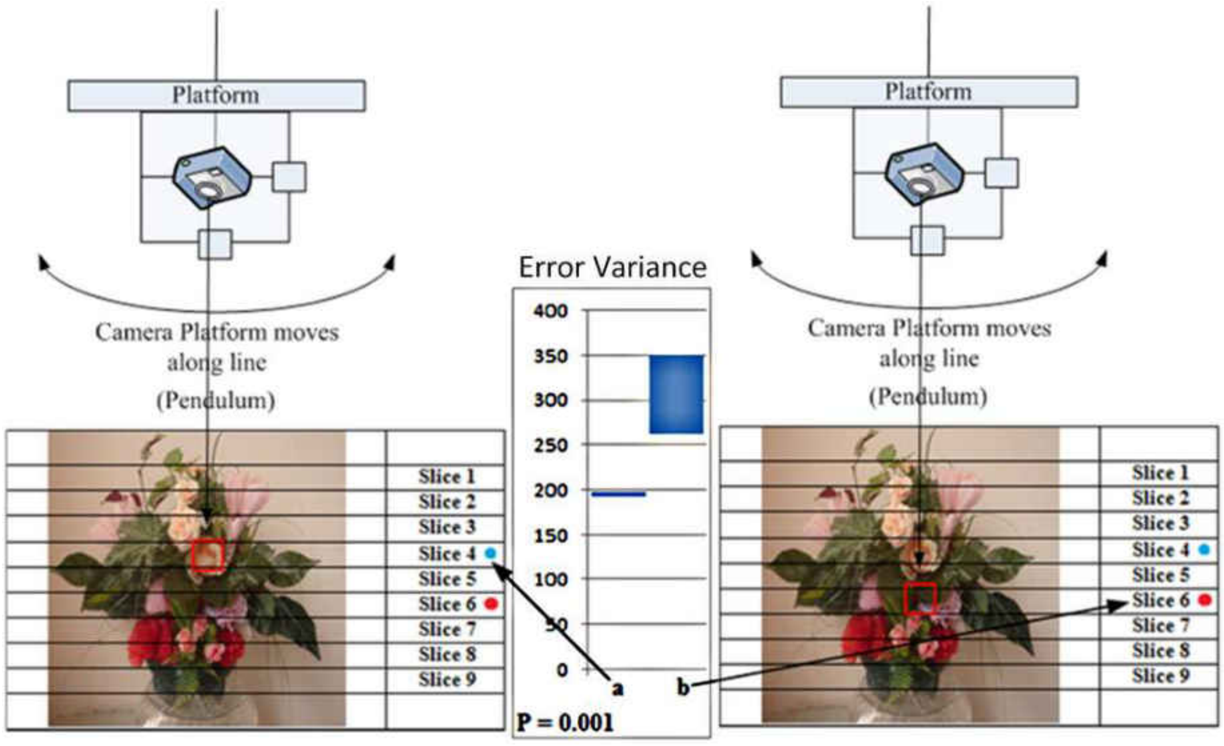


Figure 4.32 – Statistic Stabilisation comparison between slice 4 and slice 6

4.7 Conclusion

Having good knowledge about the image's properties is the main prerequisite which leads to knowing where is best to locate the reference patch. As earlier discussed, being able to locate the patch in the right position will help to generate a good error surface or error curve, which would help to track the right position. A strategy to use the generated surfaces (or curves) is needed to reduce the time consumption and increase the system's performance.

Our work targeted those issues by discussing why a larger patch is better in generating better error surfaces (or curves) and on the other hand, why a too large patch can sometimes be a disadvantage. This is because it will lose the ability to track the patch in a high speed manoeuvre (see Chapter 4). Therefore, choosing the (*100* × *100*) patch's size was the outcome of this compromise. Also, being able to track the view means the reference patch can still be found within the image space and locating the patch in the central part gives equal opportunities for the patch to be found on either the negative or positive directions of the X axis. Figure 4.33 demonstrates the difference in possible iterations that the patch can have on both sides of the central X axis. During the view tracking process, the left side image, where the patch was captured from the most central part, will have the opportunity to move and track the reference patch to both sides of the X axis equally. However, as the right side image demonstrates, if the patch is located to be somewhere other than the central part then we will face less possibilities of tracking on one side and more on the other side, hence giving advantages on one side and disadvantages on the other. Therefore, to give both sides equal opportunities, we decided to locate the patch on the central part of the X axis.

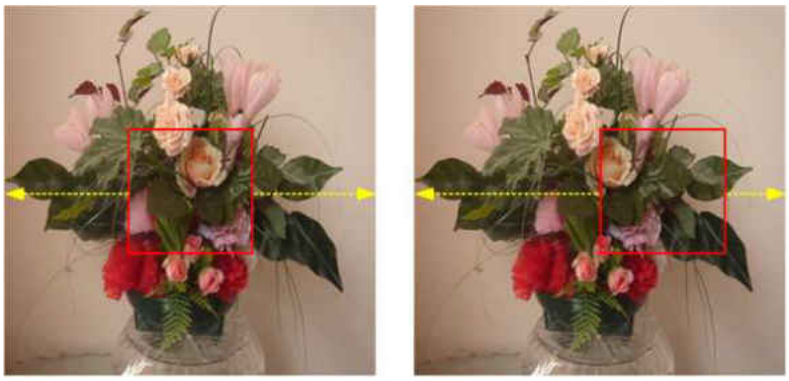


Figure 4.33 – Locating the patch in the most central position gives equal tracking opportunity to both sides

After deciding to locate the patch in the most central part of the X axis we needed a methodology which tells us where is best to locate our patch over the Y axis. This is a highly important process as the result will affect the ultimate outcome (Stabilisation). The Slice Selection Process (SSP) was designed to analyse the image's view and provides us with the answer to where is best to locate our patch over the Y axis. SSP is the major contribution in this work and experiments show how it can have an influence over the stabilisation process.

Therefore, as our question in chapter 1 requires, SSP would resolve the problem of defining the appropriate regions for image-based stabilisation using pixel-wise comparison.

Next two chapters discuss the experiments in both the laboratory and real world environments to increase the credibility of our designed SSP method in real time practice.

Chapter 5

Laboratory Experiments

5.1 Preface

This chapter combines the approaches from the previous chapters and tests them in a laboratory environment. The SSP is used in combination with the hierarchical search described in Chapter 4.

5.2 Experimental Images

To start applying the tests, we have added five extra images to those shown in Figure 4.25 (see Chapter 4) in order to introduce more variations and to improve the quality of the results. Choosing these images was based on the different characteristic each image has which means each image has satisfactory different properties compared to the other nine images. For instance, the Garden image was used to test the performance over the green areas where the Door and Building images use the highly repetitive feature alongside the image. Therefore, each image was used to test the platform's performance over different attributes. However, we could design artificial images for the same purpose, but using real world images would allow us to observe the platform's performance much closer to its real world performance. The purpose of this experiment is to examine the relations between the SSP and the stabilisation technique. The slice with the lowest DC is most likely to indicate the best slice and the highest will be the worst to use for stabilisation. The lowest number of DCs amongst the whole slices in the same image is most likely to be the best or one of the best slices on which to stabilise the camera. However, as with Patch Slice Selecion (PSS) process (see 4.3), because of the time consuming, nature of our working environment (consistent condition) and condition of our lighting resulted to have the decision to run single experiment on the slice selection process but, the stabilisation technique is repeated five times over the best and worst images' slices. Figure 5.1 includes all 10 images [115].

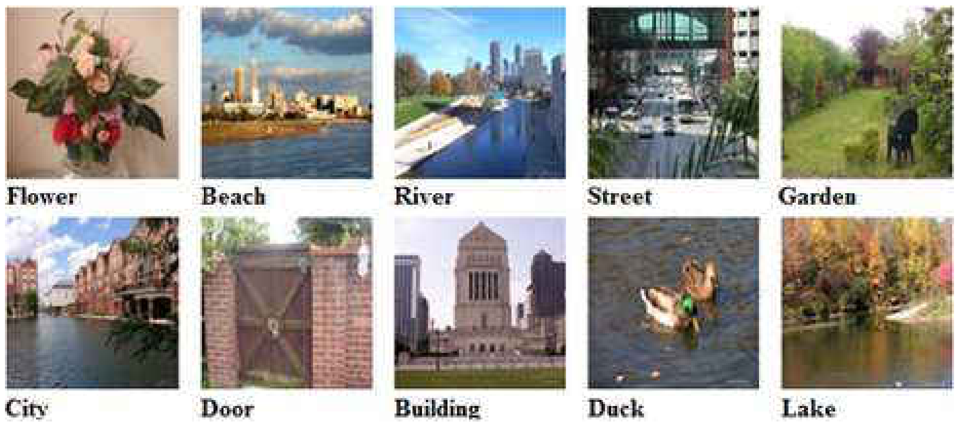


Figure 5.1 – Flower, Beach, River Street, Garden, City, Door, Building, Duck and Lake images to be printed and placed below the hanging platform

5.3 SSP Analysis

As applied earlier, we re-apply the process with the above images by hanging the platform to be pointed downwards to the printed version of the above images and apply the SSP method to determine the best slice for each of the images shown. Table 5.1 shows the results from the SSP over the 10 images. Due to having quite stable environmental conditions, the best and worst slices shown in Table 5.1 were only retrived from one run experiment.

Table 5.1 – Best and worst slices of the ten images

Image	Flower	City	Garden	River	Door	Beach	Lake	Building	Street	Duck
Best Slice	4	4	1	6	6	6	3	4	3	5
Best DC	87	82	138	87	188	75	74	129	88	76
Worst Slice	6	9	6	1	3	9	9	1	9	9
Worst DC	309	548	344	606	412	562	526	312	243	436

Figure 5.2 shows the long vertical patch which will be divided into 9 parts for the purpose of processing the SSP method. A few images have gained more than one slice with low DC which

shares an equivalent value. Therefore, we will be using one of the slices with a low DC number by random selection (i.e. Building image). However, slices with a low DC are titled “a” and those with a high DC are titled “b”.

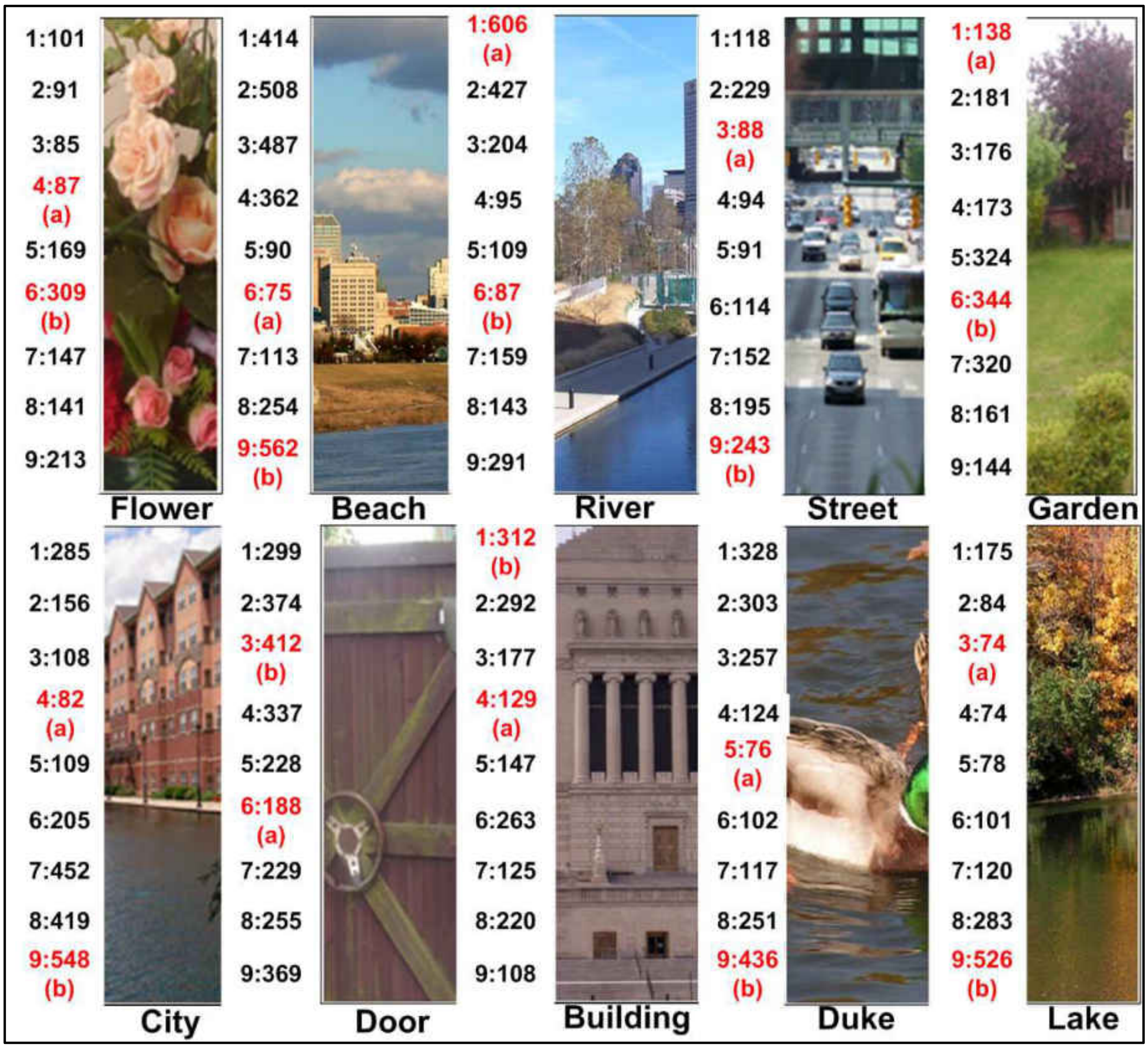


Figure 5.2 – DC values for all 10 images slices

Figures 5.3 to 5.12 show the best and worst slices besides their Pixel Displacement Graphs retrieved from applying the SSP technique over the 10 images. We will show how the best and worst slices differ in smoothness by visually showing how the best slices have fewer noises than the worst slices. From the SSP point of view, it means the method was more successful in gaining more precise values over the best slice. We will also explain the visual appearance differences between every two slices (best and worst) on each image which were the effective factors in shaping the pixel displacement graphs. Though, all the best slices are titled “a” and the worst “b”.

Starting with the Beach best and worst slices (see Figure 5.3), the SSP method gained quite a low DC over the best slice. The visual differences between the two slices are obvious where the wide variety of features (i.e. buildings, trees, sky and etc…) on slice “a” is observable but the water has dominated most of the view on slice “b”. This water domination made large colour constancy across the entire slice and forced the SSP to generate a high DC pixel displacement graph.

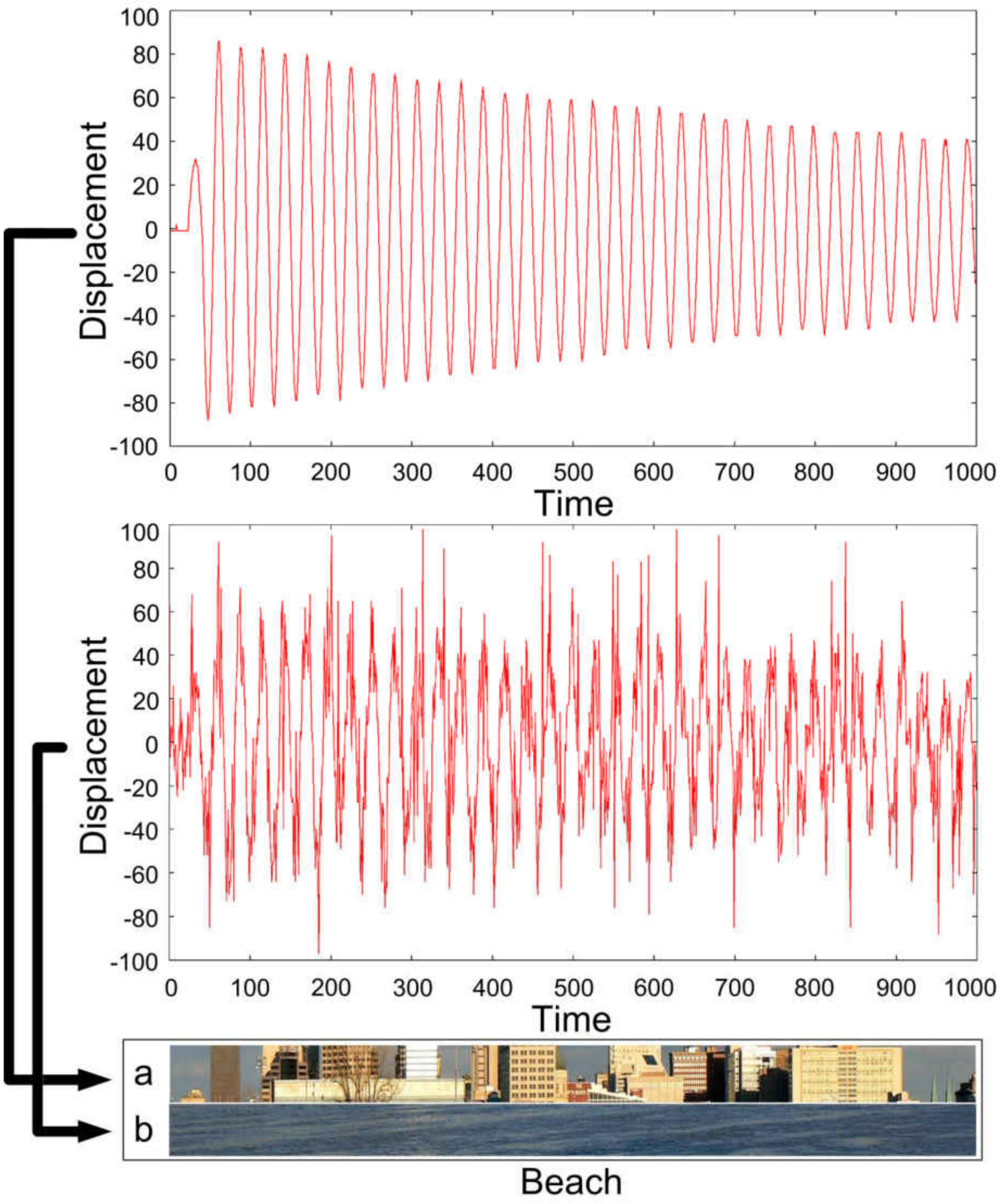


Figure 5.3 - Best and worst Pixel Displacement Graphs for the Beach image

On the City image, it shows the water has dominated the majority of the slice "b" view. This caused the appearance comparison between consecutive images to introduce numerous similarities which caused a high DC value on the retrieved displacement graph. The scenario differs with slice “a”, where different features (i.e. buildings, sky and green leaves) caused high colours/regions variations. These variations give the SSP method the opportunity to retrieve more precise displacement values and generate a displacement graph with a lower DC value (see Figure 5.4).

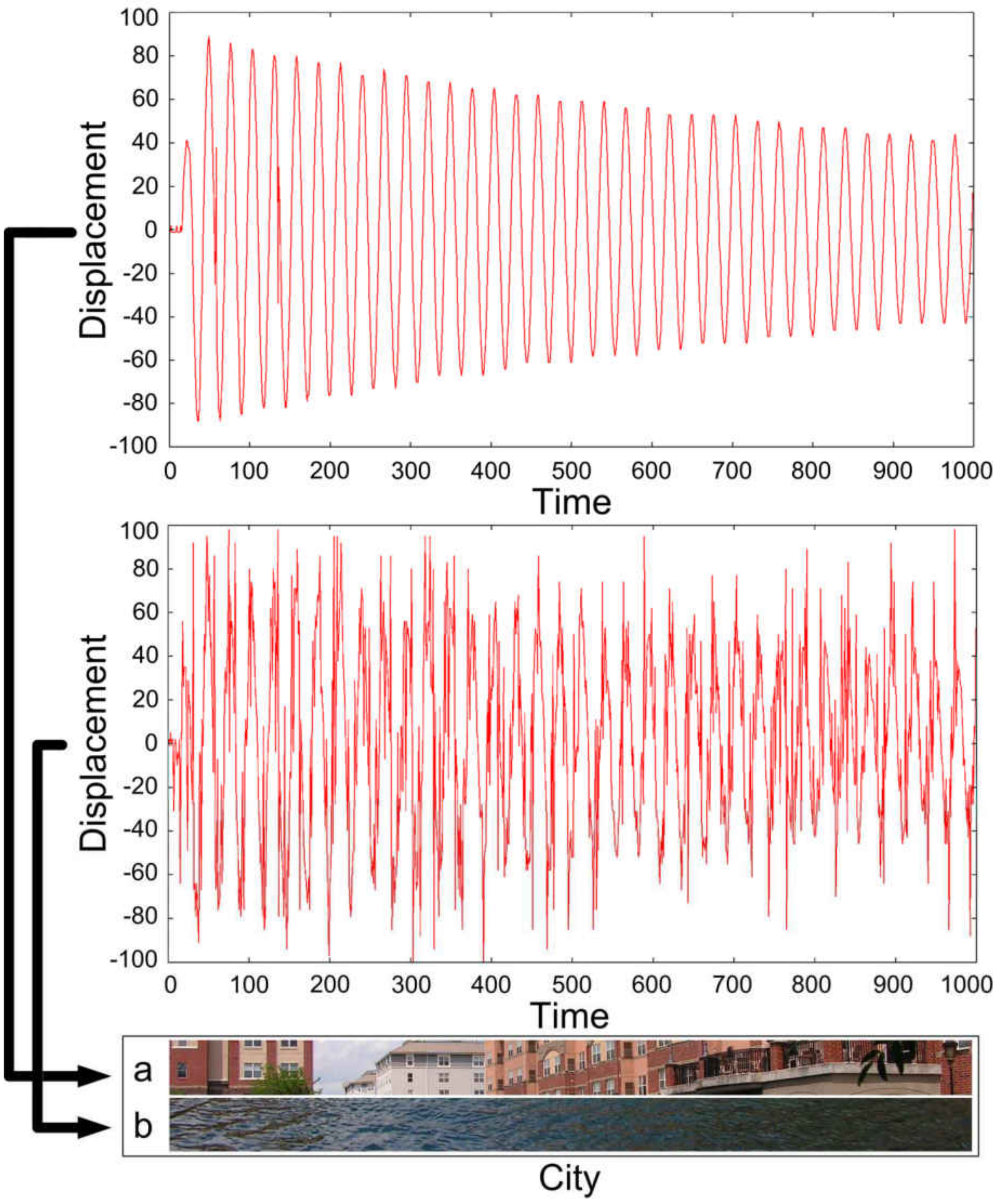


Figure 5.4 - Best and worst Pixel Displacement Graphs for the City image

The region uniqueness in Figure 5.5 (Door best and worst slices) is significantly missing, therefore the difference between the best and worst slice is the door's handle and some side features only. However, even the door handle is not a highly distinctive feature and we cannot rely on it to expect a good SSP outcome. Although, due to gaining 188 DC over slice “a”, the best slice may still not be good enough from the stabilisation technique point of view. The brown colour is highly dominating across the image where the door is painted dark brown and the bricks are light brown. They are still two different colours but any blurriness will increase the similarities between the two colours levelling (dark and light brown). Though the white lamp and the green leaves are highly distinctive features appearing on the left and right hand sides of slice “a”, as mentioned earlier, the pendulum’s speed is higher in the central and the reference patch is captured from the central region. Therefore, having the region uniqueness on the sides may not be highly valuable.

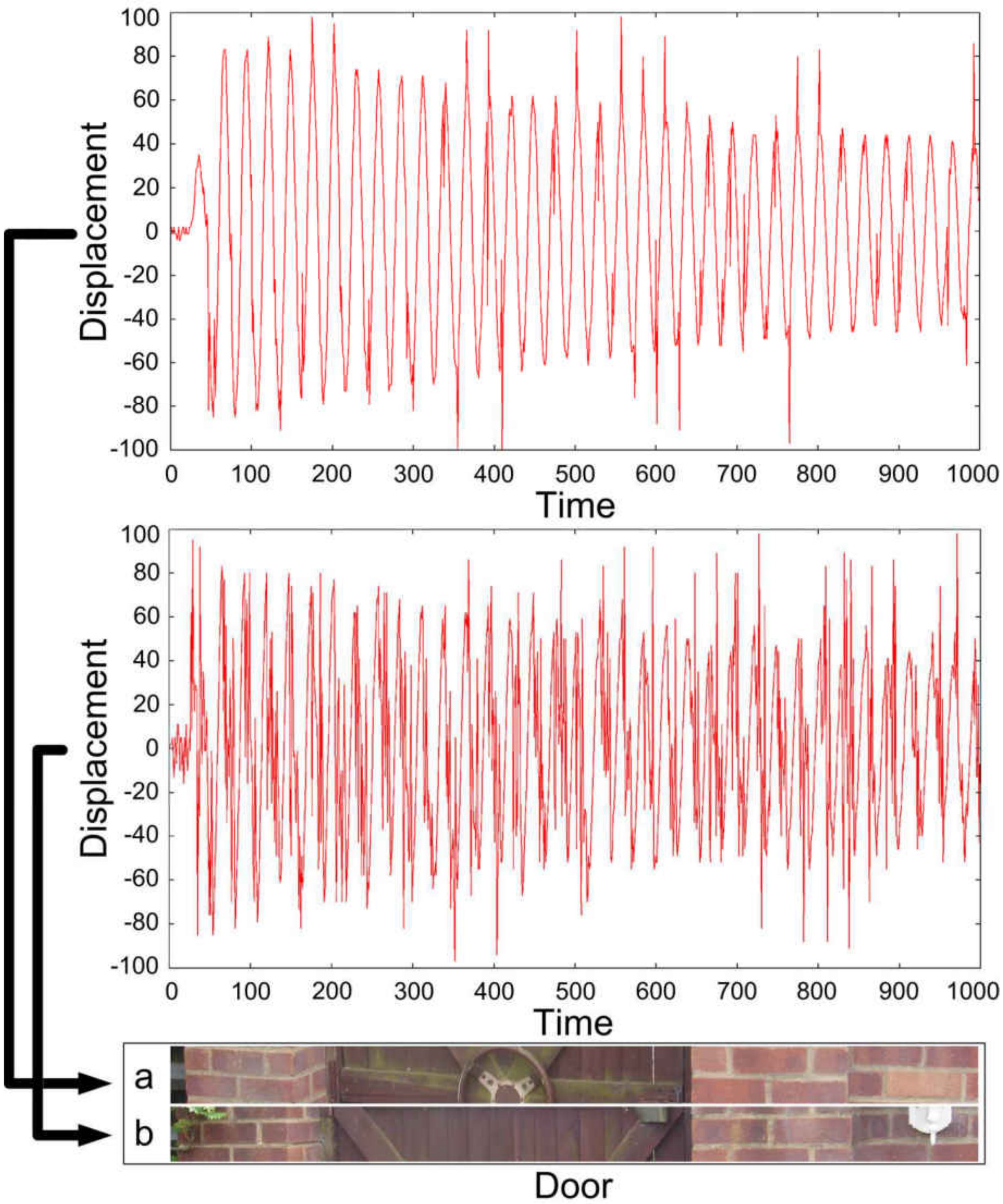


Figure 5.5 - Best and worst Pixel Displacement Graphs for the Door image

Figure 5.6 shows the displacement graphs and the best and worst slices for the flower image. It shows that the green colour on slice “b” was highly dominating, but the scenario differs with slice “a”, where the yellow flower in the central location introduced more region uniqueness. However, we may miss this region uniqueness if a smaller patch size was used (i.e. 50×50). The spatial positioning is highly important in determining the uniqueness of the captured patch, for example moving the yellow flower from left to right or vice versa makes the two patches differ highly, therefore the region’s uniqueness amongst the neighbouring regions gives us large potentiality from the stabilisation scenario point of view. The scenario with slice “b” slightly differs where some region similarities appear across the slice. This similarity can be notified by having the two pink flowers with surrounding green leaves in two locations of the same slice. This makes the region uniqueness less possible and increases the overshoots during the slice selection procedure.

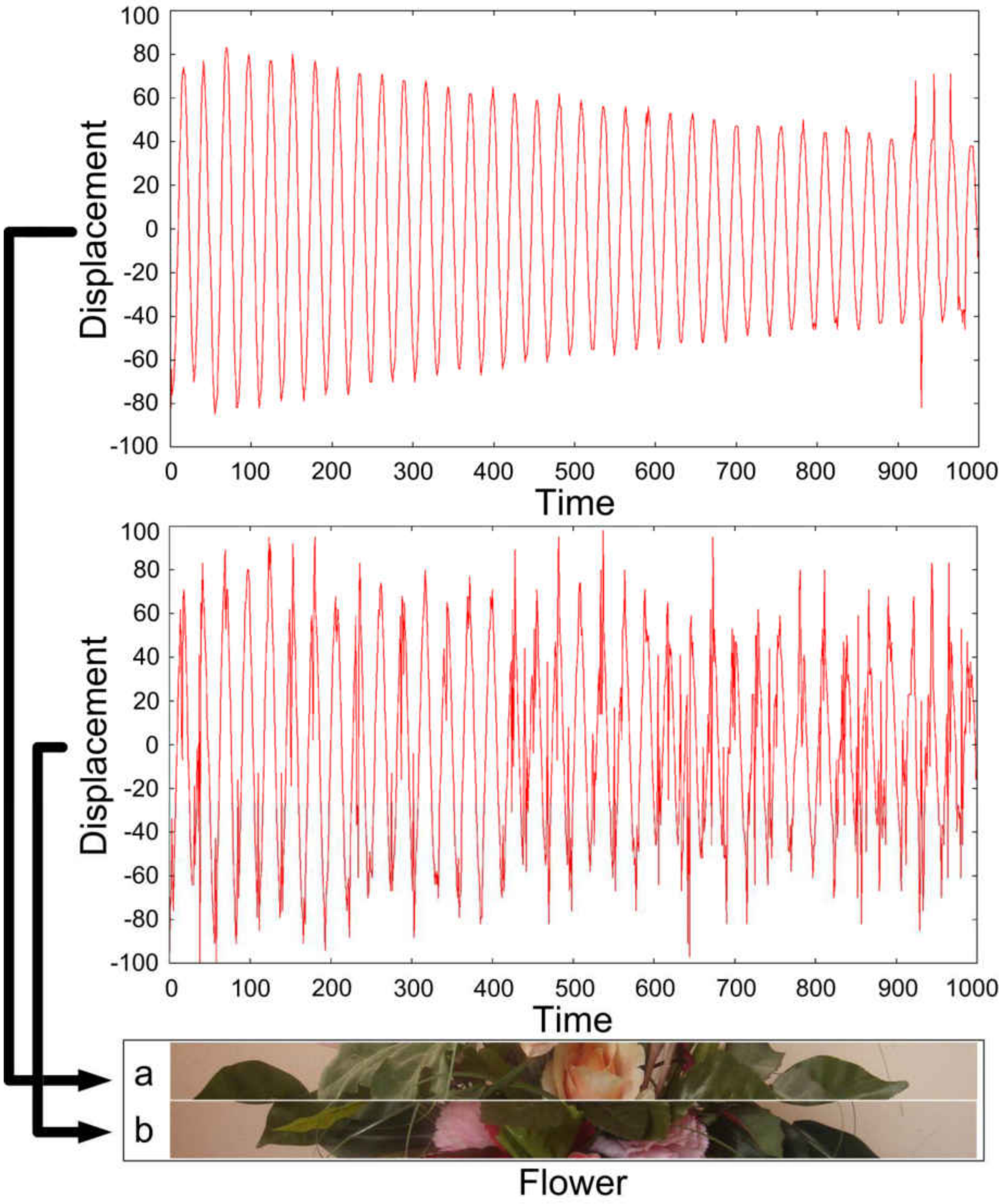


Figure 5.6 - Best and worst Pixel Displacement Graphs for the Flower image

Figure 5.7 shows how the best and worst slices differ in the way their colour contents are distributed. The central region of slice “a” is highly distinctive by capturing a red leafed tree within the central region and the sky’s views on both neighbouring sides. Both sides of slice “a” are dominating with green leaves but they differ in how these leaves are distributed. These differences will have a significant influence on the SSP method. The right hand side leaves are more saturated than the left hand side green leaves. Moreover, in contrast to the right hand side, the left hand side leaves are more clustered where no sky views are visible through some of the parts. In slice “b”, the green grass is the dominating view within the central and the left hand side regions. However, as the pendulum’s speed is higher in the central than in anywhere else, this domination will increase the over shoots of the SSP displacement graphs. It would also give us a clear understanding of why the overshoots appear more in graph “b” than in “a”.

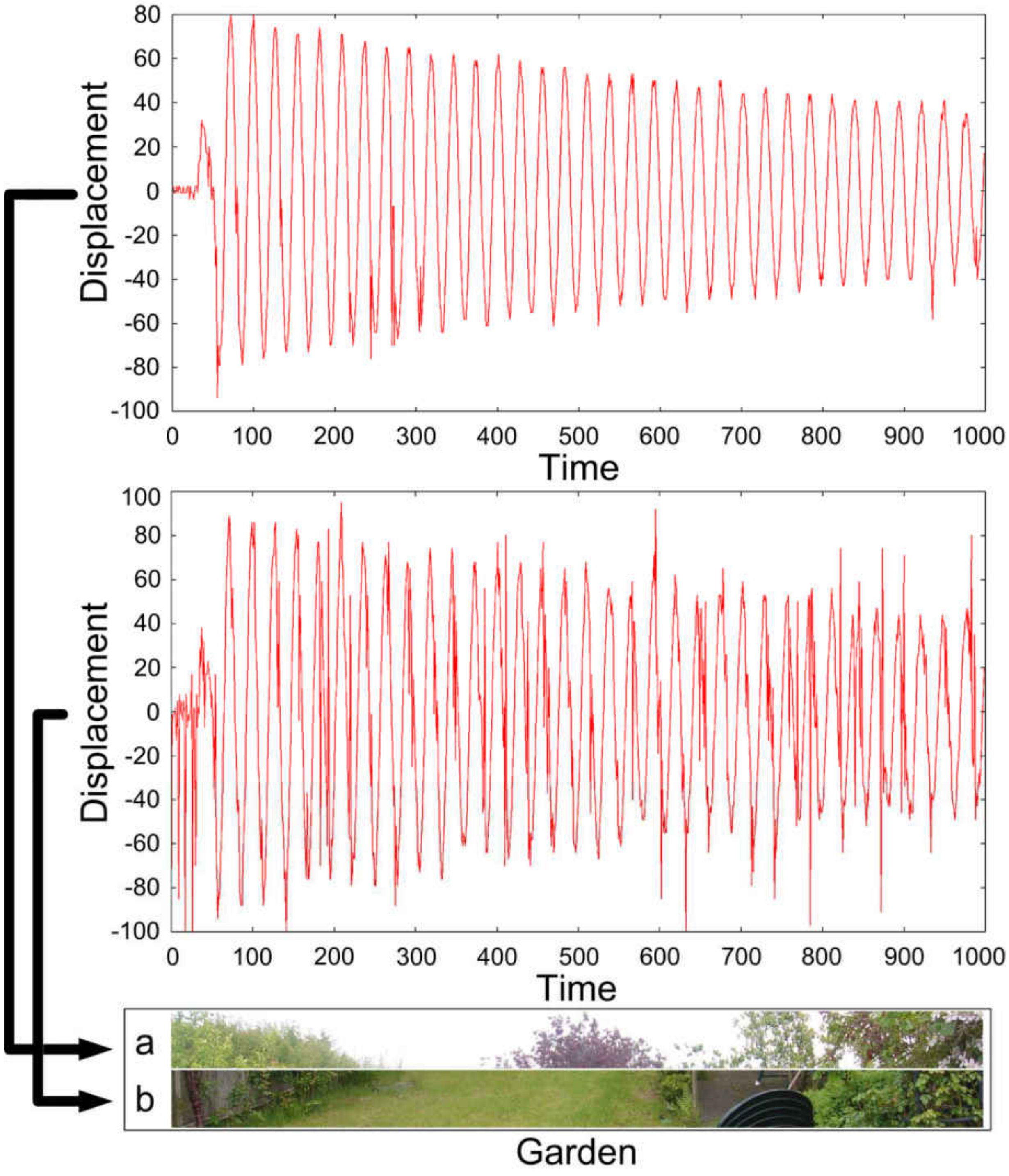


Figure 5.7 - Best and worst Pixel Displacement Graphs for the Garden image

Looking at Figure 5.8, the single colour domination has also appeared on the Building image, where grey is the dominating colour in the central chunk of both best and worst slices. Also, the sky's view conquered a large part of both slices which introduced further colour constancy across the slice. Slice “a” has a larger sky view than slice “b” but slice “b” looks quite similar to the Garden slice “a” scenario where the central region is a unique triangle shape surrounded by white sky’s views in both neighbouring regions. Moreover, the contrast comes with having the building’s peak as the only variation seen to the slice but in the Garden image, few variations were on the sides that had an influence on the ultimate SSP outcome. Also, the features on the Garden “a” slice had more saturated colours (i.e. red and green), but in the Building scenario the building’s peak was grey and any blurriness can increase the similarities between the Building and the background sky view (white colour). Also, slice “a” on the Building image contains further feature variations to the sides which caused the SSP method to gain a lower DC value.

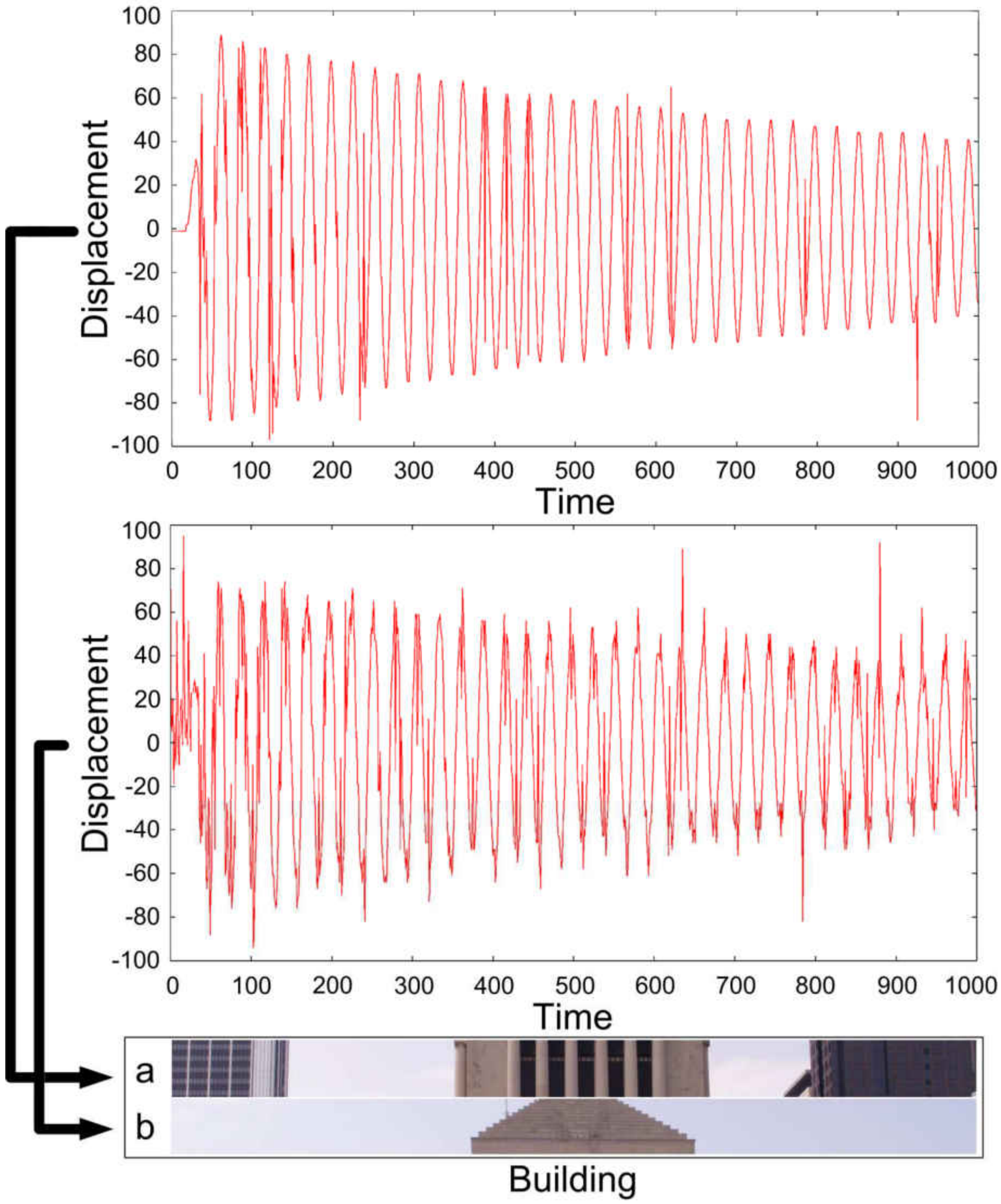


Figure 5.8 - Best and worst Pixel Displacement Graphs for the Building image

Slice "b" on the River image shows the sky's view and the light blue colour is highly dominating. We do have a few white clouds which may break this colour constancy, but encountering little blurriness may introduce further similarities and cause degrading on the tracking performance. Slice “b” contains good colours/features variations, while if we divide the slice into chunks, the left chunk is a grass view and the right is a wall with grey colour bricks. However, the grass colour constancy was also ceased by a large shadow. The central region of this slice includes good feature variation (i.e. canal, side roads, trees and etc…). All these variations, from the sides to the central regions, made this slice to be highly useable from the SSP method point of view. The constancy on slice “b” and the variations on slice “a” were reflected in how the pixel displacement graphs were generated (see Figure 5.9 (River)).

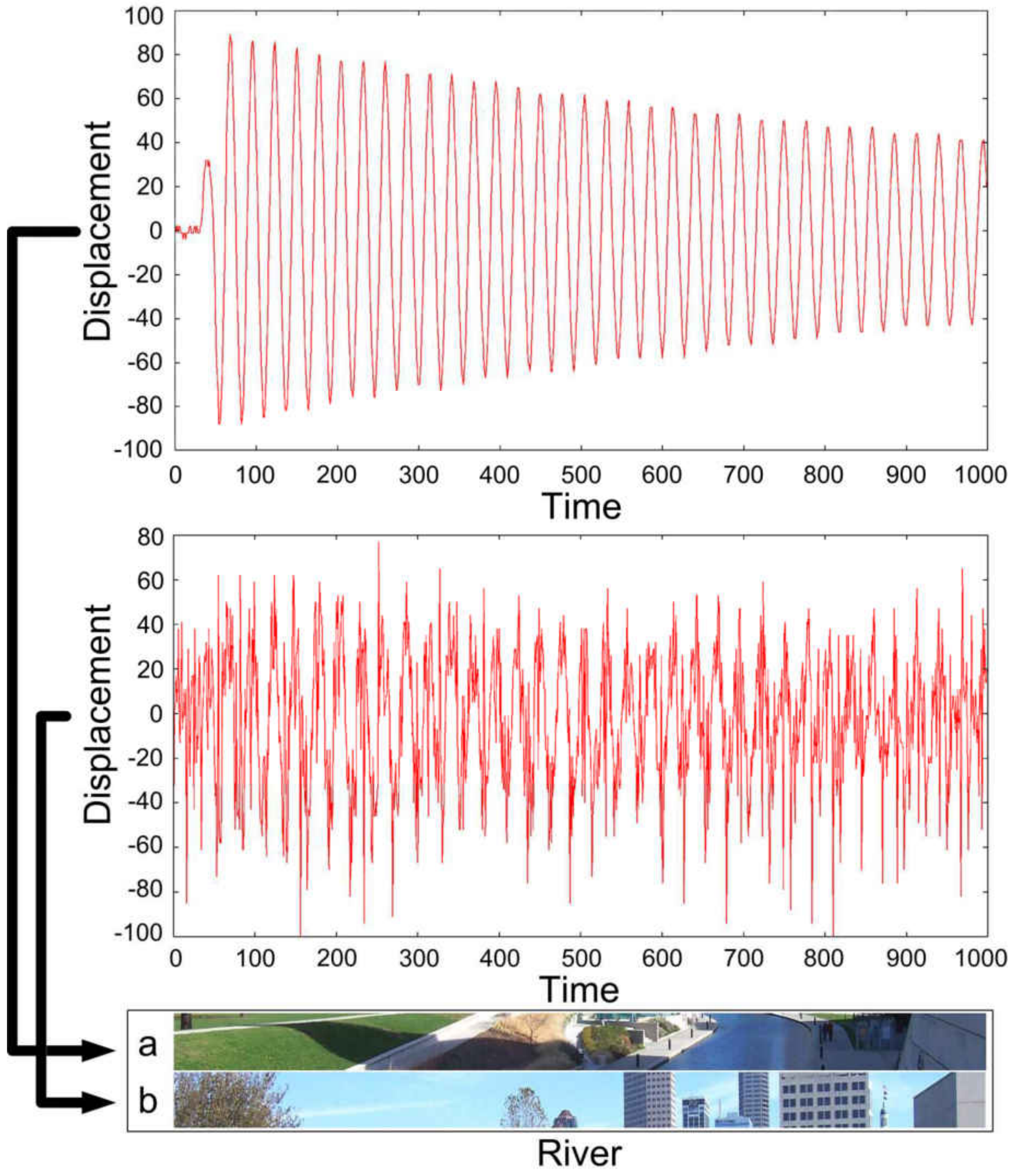


Figure 5.9 - Best and worst Pixel Displacement Graphs for the River image

As with previous examples (i.e. River, Beach), slice "b" on the Duck image (see Figure 5.10) includes high colour repetition. This repetition is caused by the domination of the water's view along the slice. In contrast, slice "a" includes highly saturated colours where the ducks and few dark patches are located across the slice. This encouraged the SSP to determine many precise displacement values over this slice.

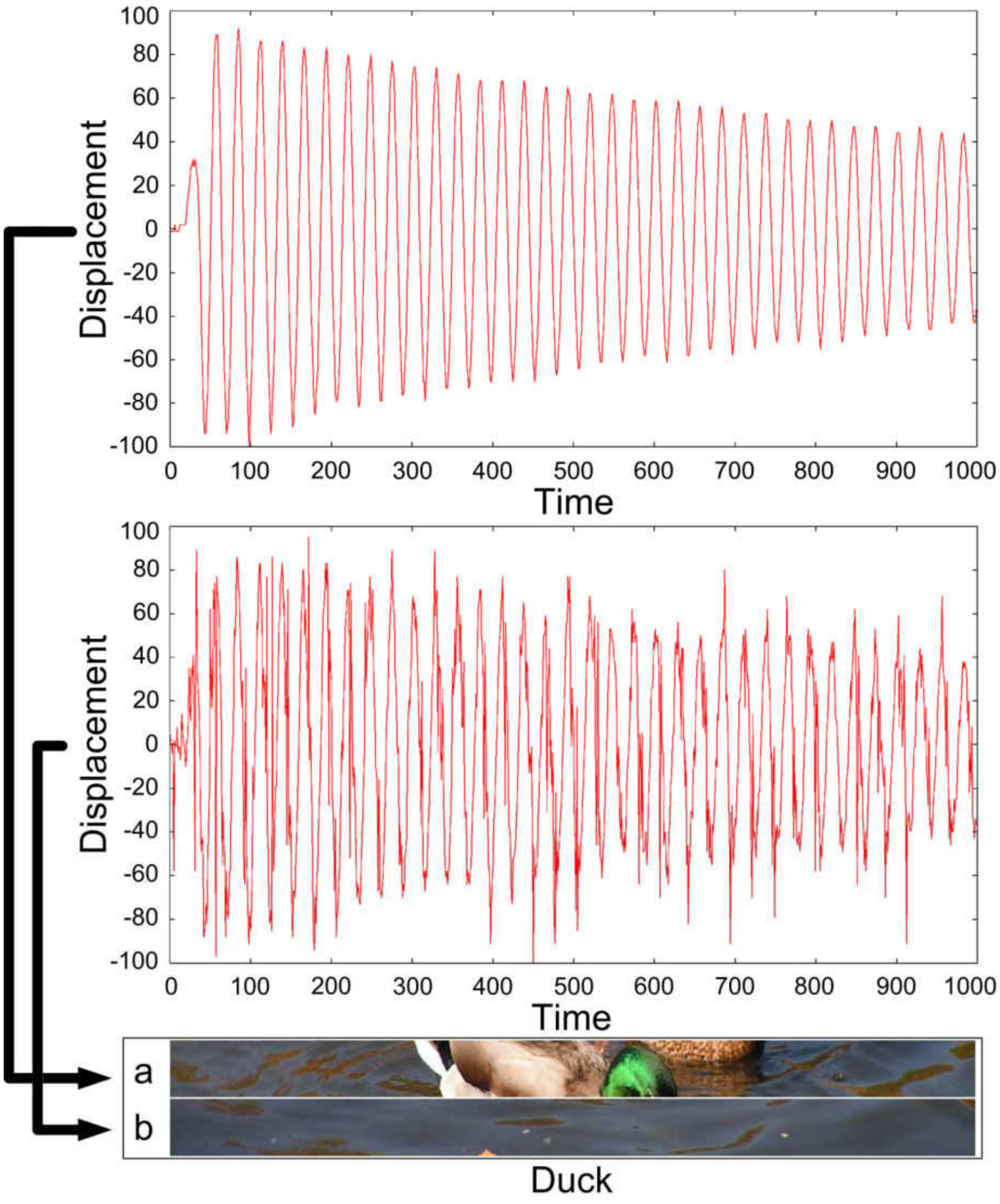


Figure 5.10 - Best and worst Pixel Displacement Graphs for the Duck image

Similar to Duck slice "b", Lake slice "b" was also dominated by the water's view, therefore, getting a high DC value is highly expected. Nevertheless, despite the colour constancy, the water's view can have many colour variations compared to the sky's view. In contrast to the sky's view, the colour variations on the water surface are mostly due to the water's reflection ability. We can observe some of those reflections over Lake slice "b" but due to the blurriness, those reflections are indistinct which made the SSP method fail to retrieve the precise displacement values. In contrast to slice "b", slice "a" includes massive colour variations on various tree leaves. Usually, leaves initiate vast repetitions, but in this scenario the clustered leaves introduced significant region variations across the slice.

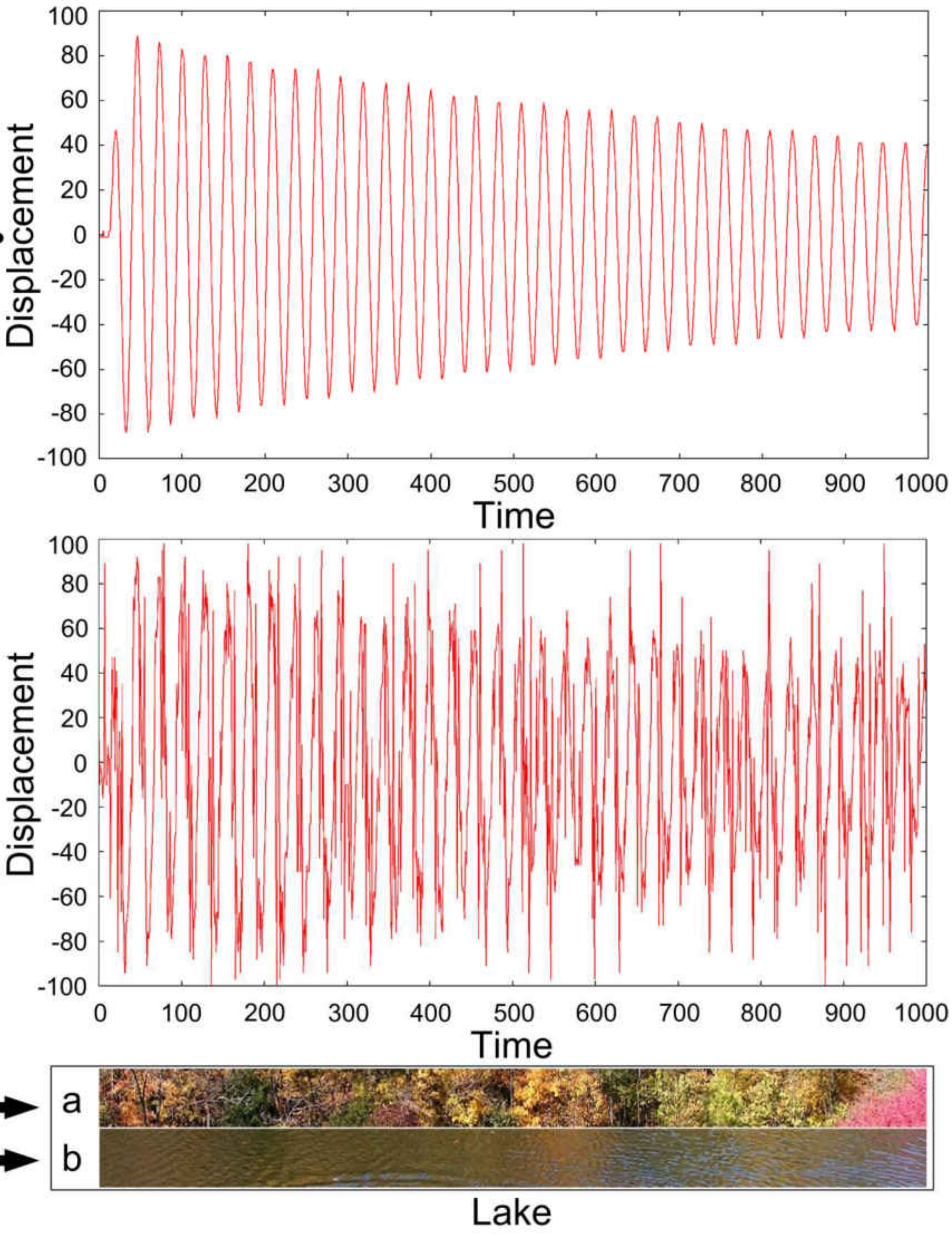


Figure 5.11 - Best and worst Pixel Displacement Graphs for the Lake image

Figure 5.12 shows the best and worst slices beside their pixel displacement graphs for the Street image. Slice “a” on the Street image shows large variations with a good region distinction. The white colour domination in the centre, gray on left and brown on the right has created a great region uniqueness across the slice. The region uniqueness on slice “b” is much less and capturing the central region as a reference causes large overshooting from the SSP method point of view. This makes the generated displacement graph gain a high DC value which indicated the non-potentiality of that slice from a stabilisation point of view.

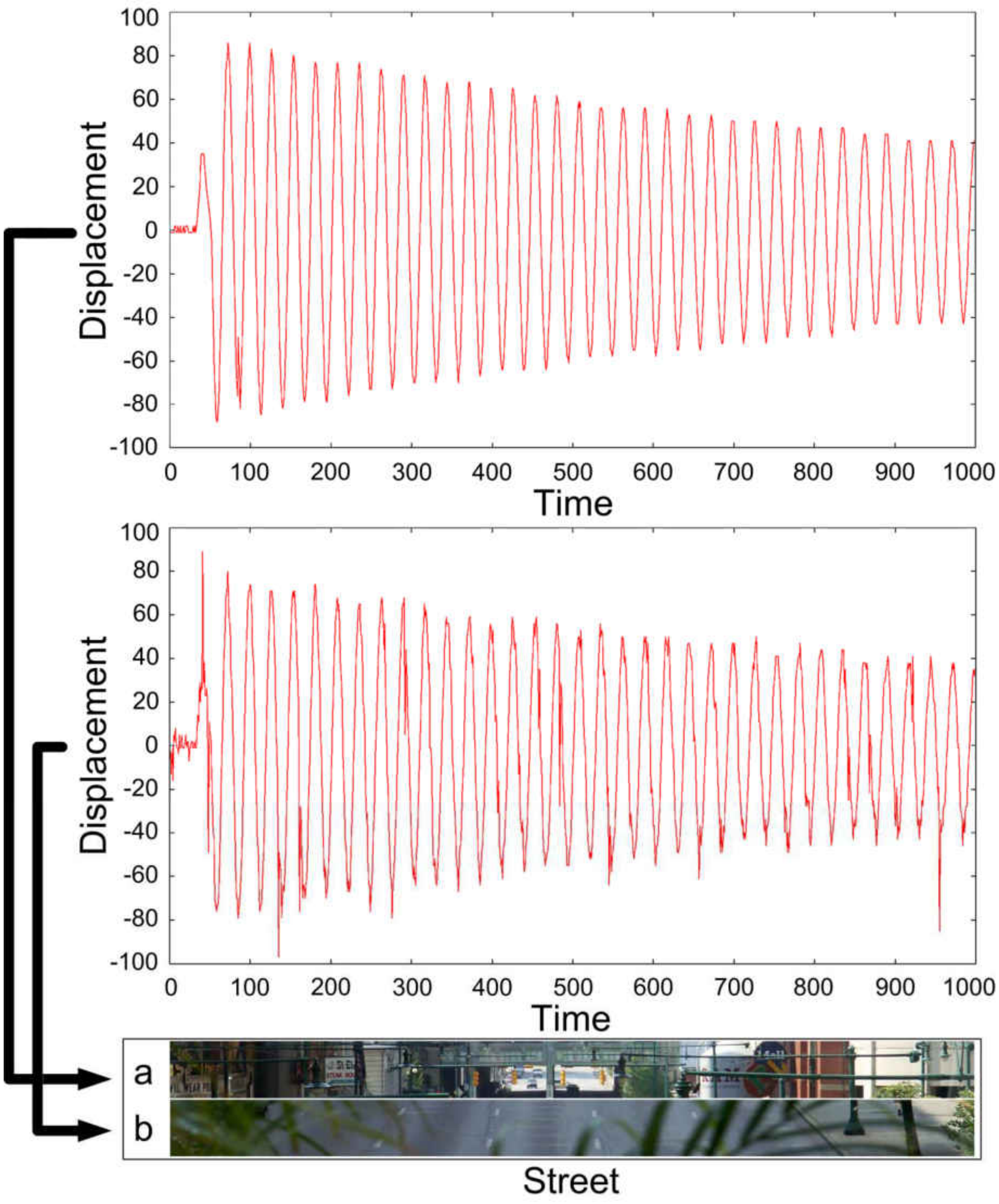


Figure 5.12 - Best and worst Pixel Displacement Graphs for the Street image

5.4 Stabilisation Techniques Using SSP Analysis

The next stage is to run the stabilisation technique over the selected slices (see Table 5.1) using one of the successful hierarchical strategies (i.e. strategy "23"), in combination with the SSP method. To evaluate the performance results, a statistical outcome (P value) was used to demonstrate the difference in performance between the best and worst slices on each image. To retrieve the P value, we have used the Kruskal-Wallis [135] technique which is a non-parametric stat supplied by the MINITAB [136] software. Also, Figures 5.13, 5.14 and 5.15 are showing the box and whisker plots of the gained data for the used images. For each image we have two separate datasets ("a" and "b") representing the stabilisation outcome over the best and the worst slices. As in Chapter 4 (see Figure 4.26), we test the 95% confidence outcome but in contrast to previous results, the success is to verify the unlikeness by getting the P value below 0.05. If the applied stabilisation on both the best and worst slices was found to be different, then it means the relationship between the SSP and the Stabilisation technique over the chosen image was successful. This success is shown by retrieving 0.009 as a P value over all 10 images.

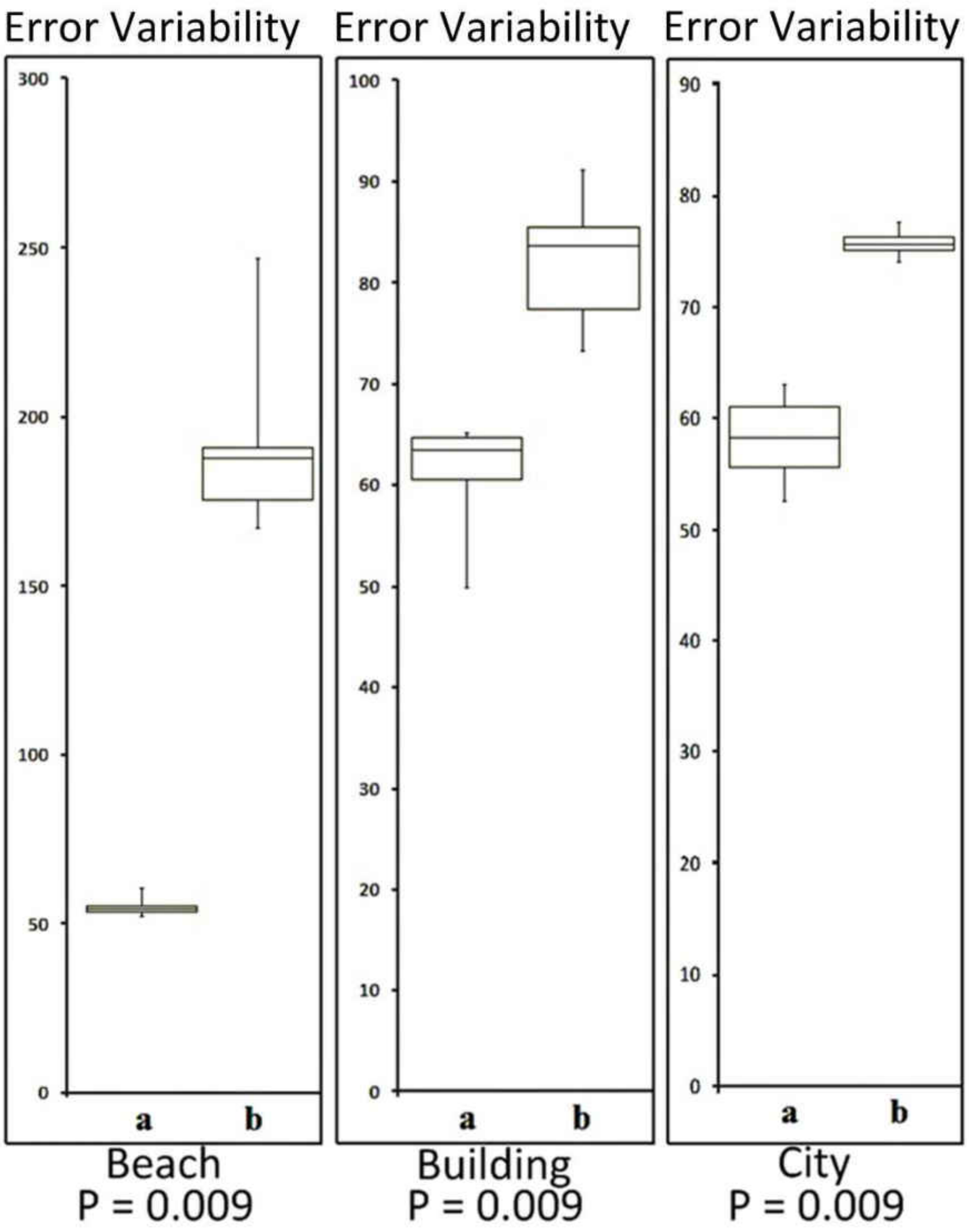


Figure 5.13 - Using box and whisker plotting to present the gained data with the retrieved P value for Beach, Building and City images

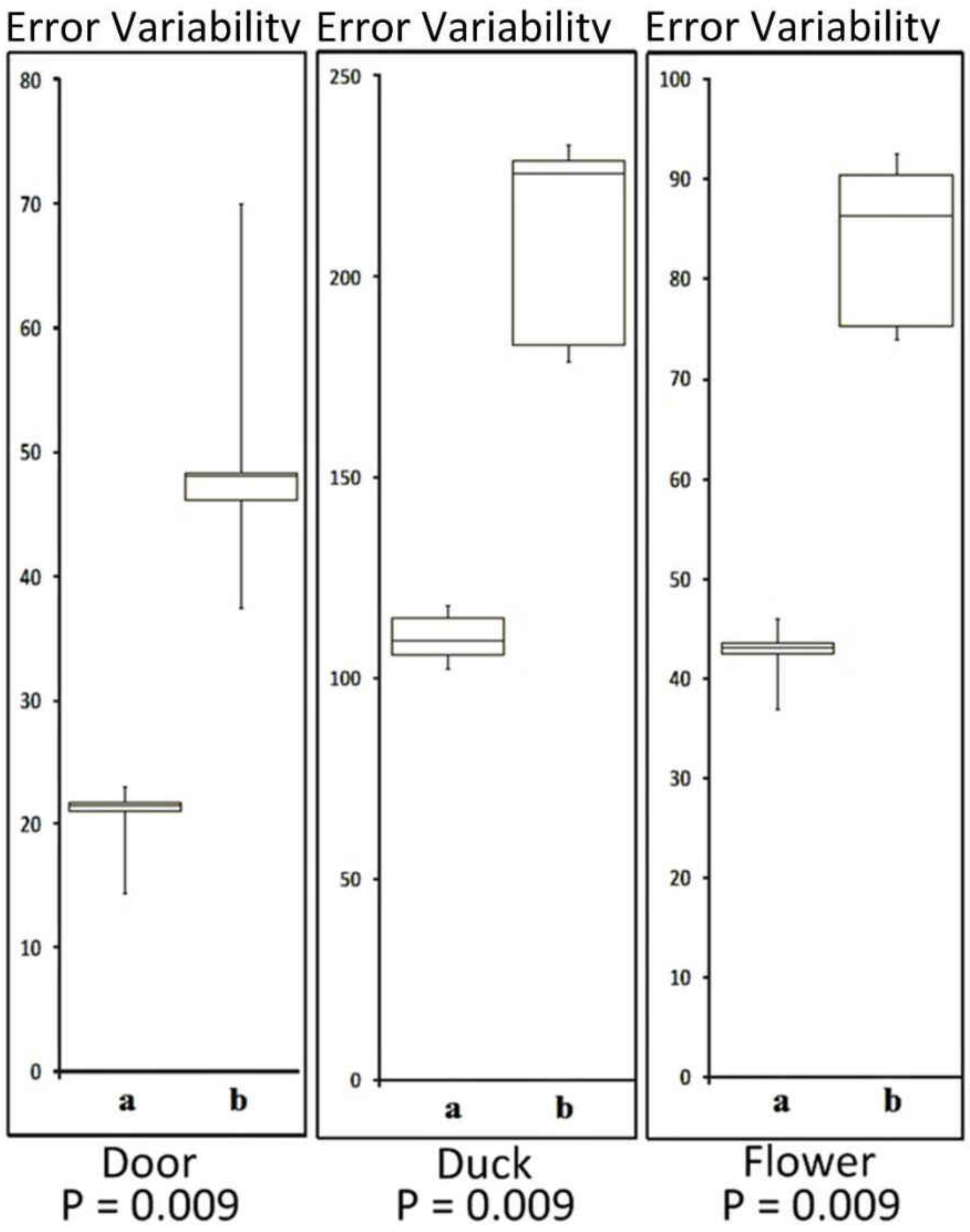


Figure 5.14 - Using box and whisker plotting to present the gained data with the retrieved P value for Door, Duck and Flower images

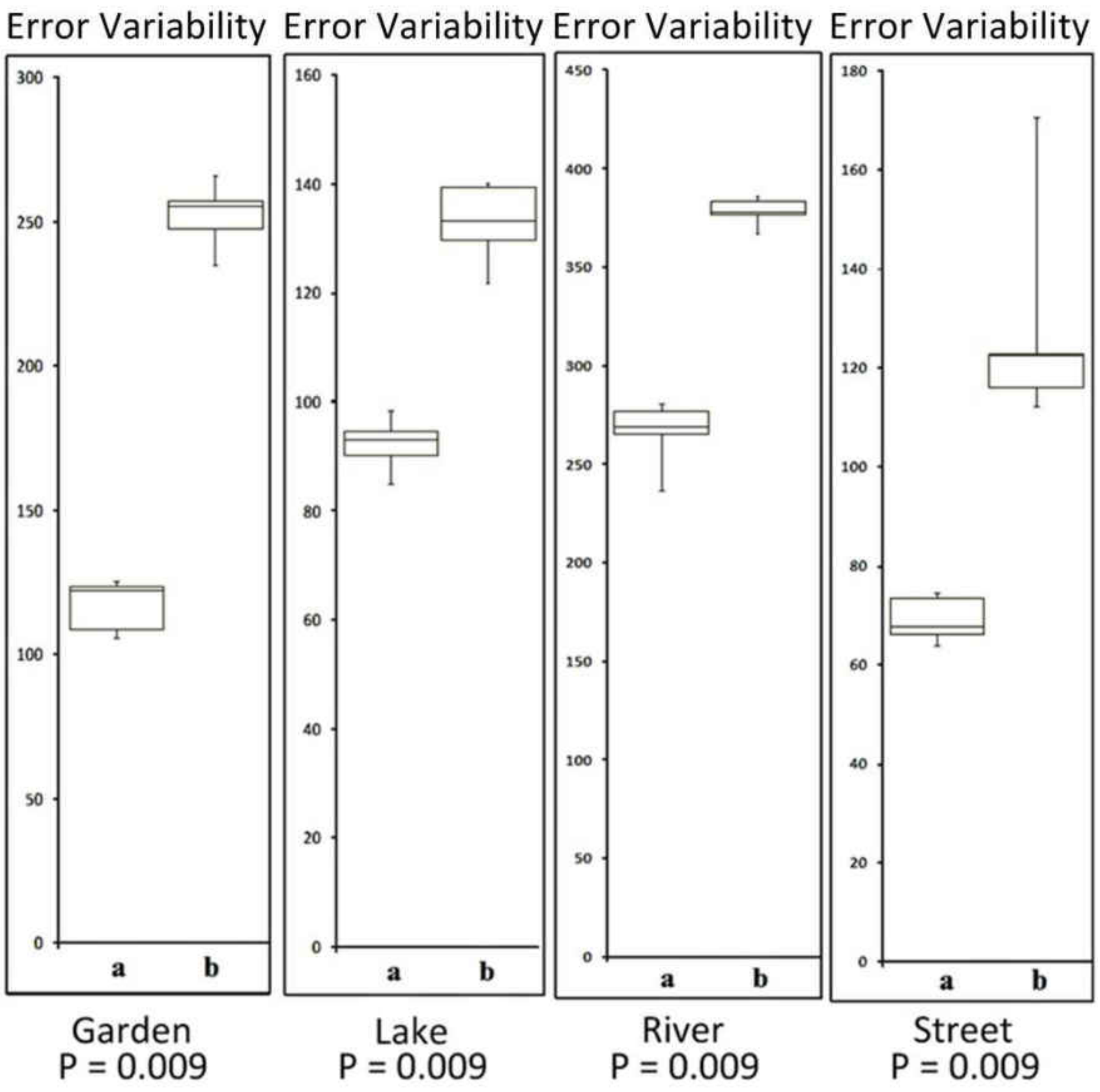


Figure 5.15 - Using box and whisker plotting to present the gained data with the retrieved P value for Garden, Lake, River and Street

5.5 Error Curves of the Selected Slices

Running the appearance comparison between the central patch and every patch across the slice provides us with a distance curve that the controller can use to navigate back to the origin. Figures 5.16, 5.17, 5.18, 5.19 and 5.20 demonstrate the defined distance curves for every best and worst slice over the 10 images. For example, the relation between the distance curve shape and the obtaining of low DCs over the Garden best slice (“a”) in Figure 5.17.a (Garden) is highly observable. In the Garden best slice, the region uniqueness is observed. The distance curve shape also demonstrates how a large descending area indicates the uniqueness of the central region against the whole slice. However, due to the colour repetition of the central region, the distance curve over the Garden worst slice (“b”) looks like it has less potential (see Figure 5.17.b (Garden)). Defining the distance curve for every best and worst slice shows a great relationship between every generated distance curve in corresponding to the generated pixel displacement graph over the same slice.

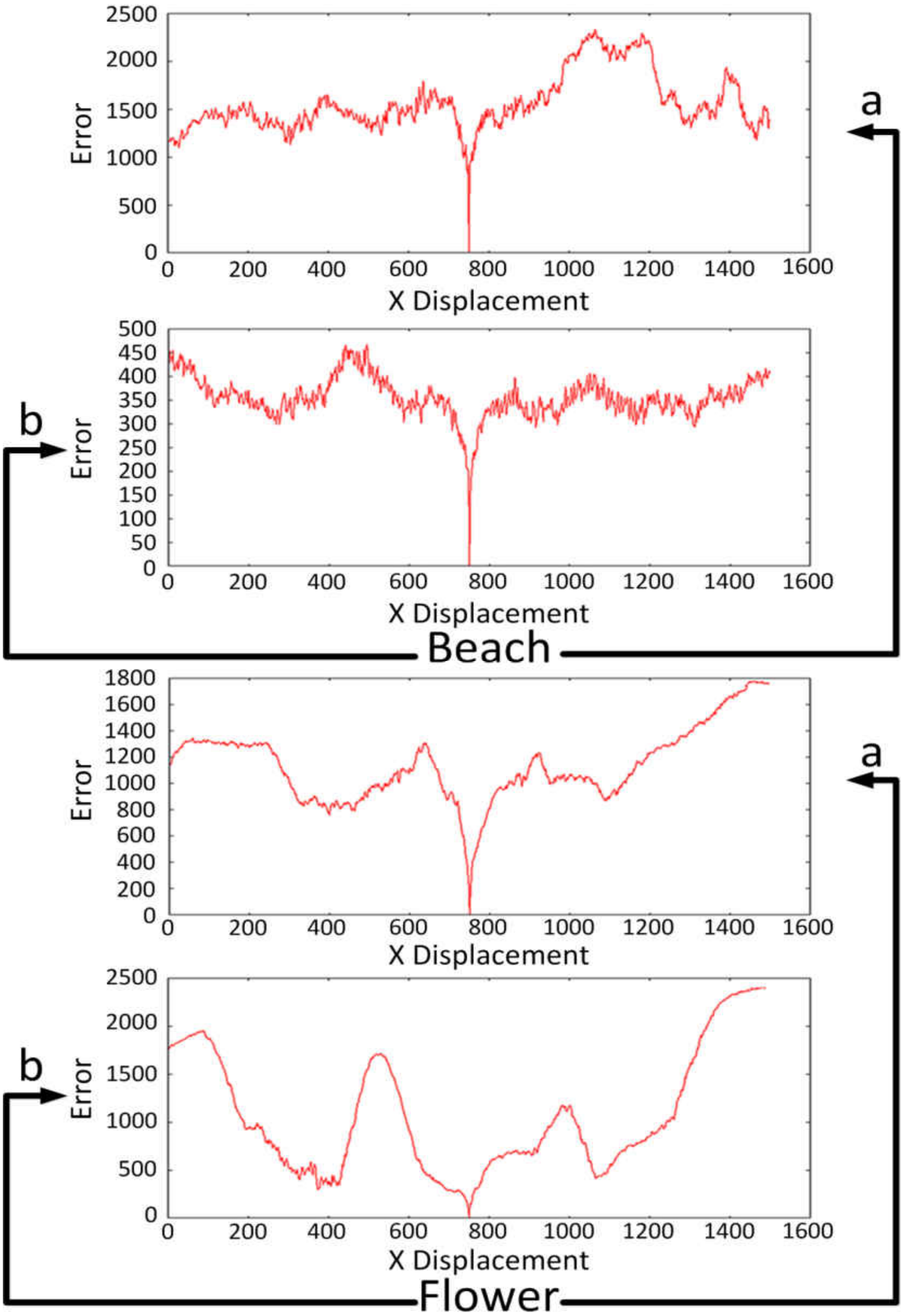


Figure 2.16 - Error curves generated over the best and worst slices of Beach and Flower images

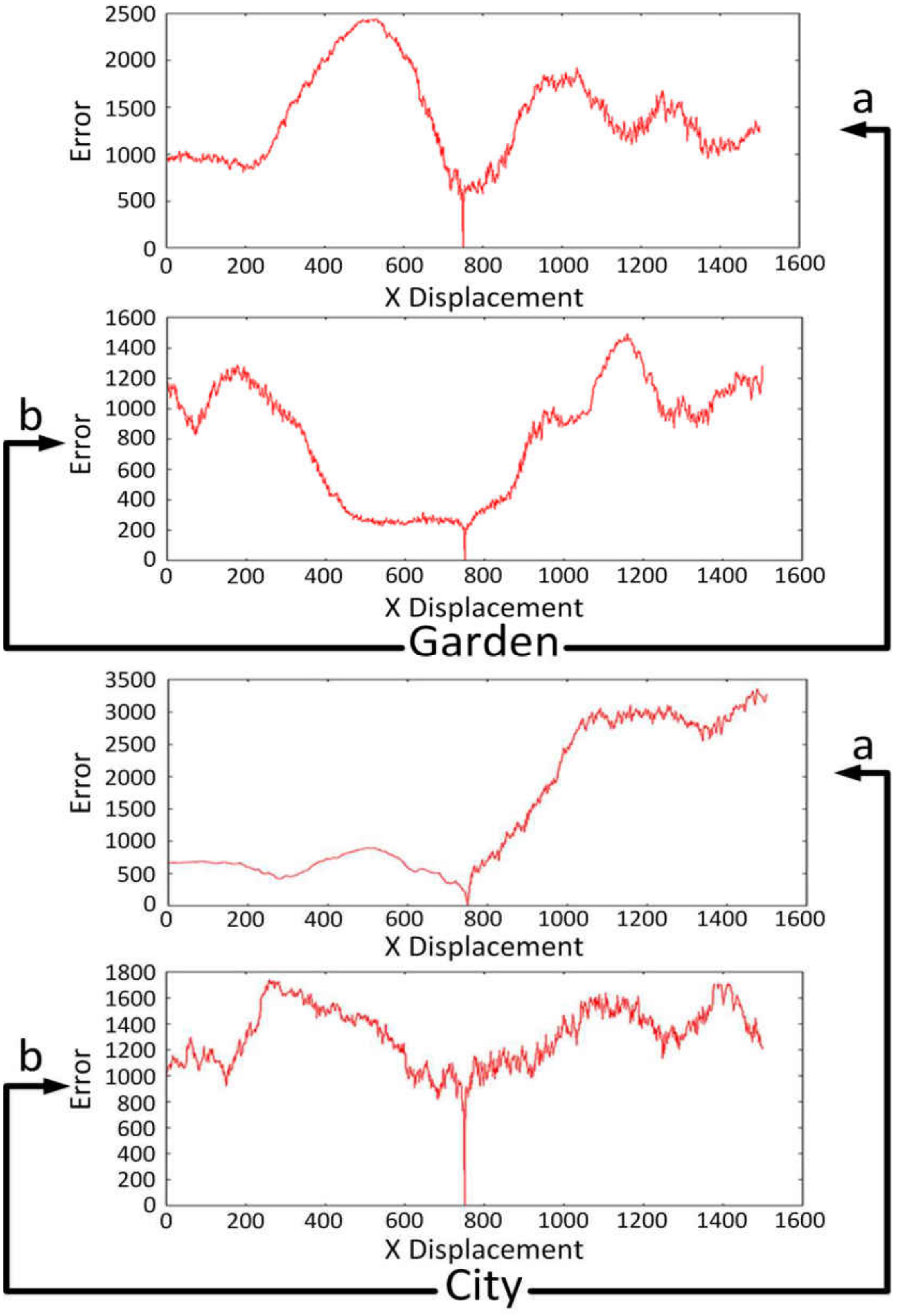


Figure 5.17 - Error curves generated over the best and worst slices of Garden and City images

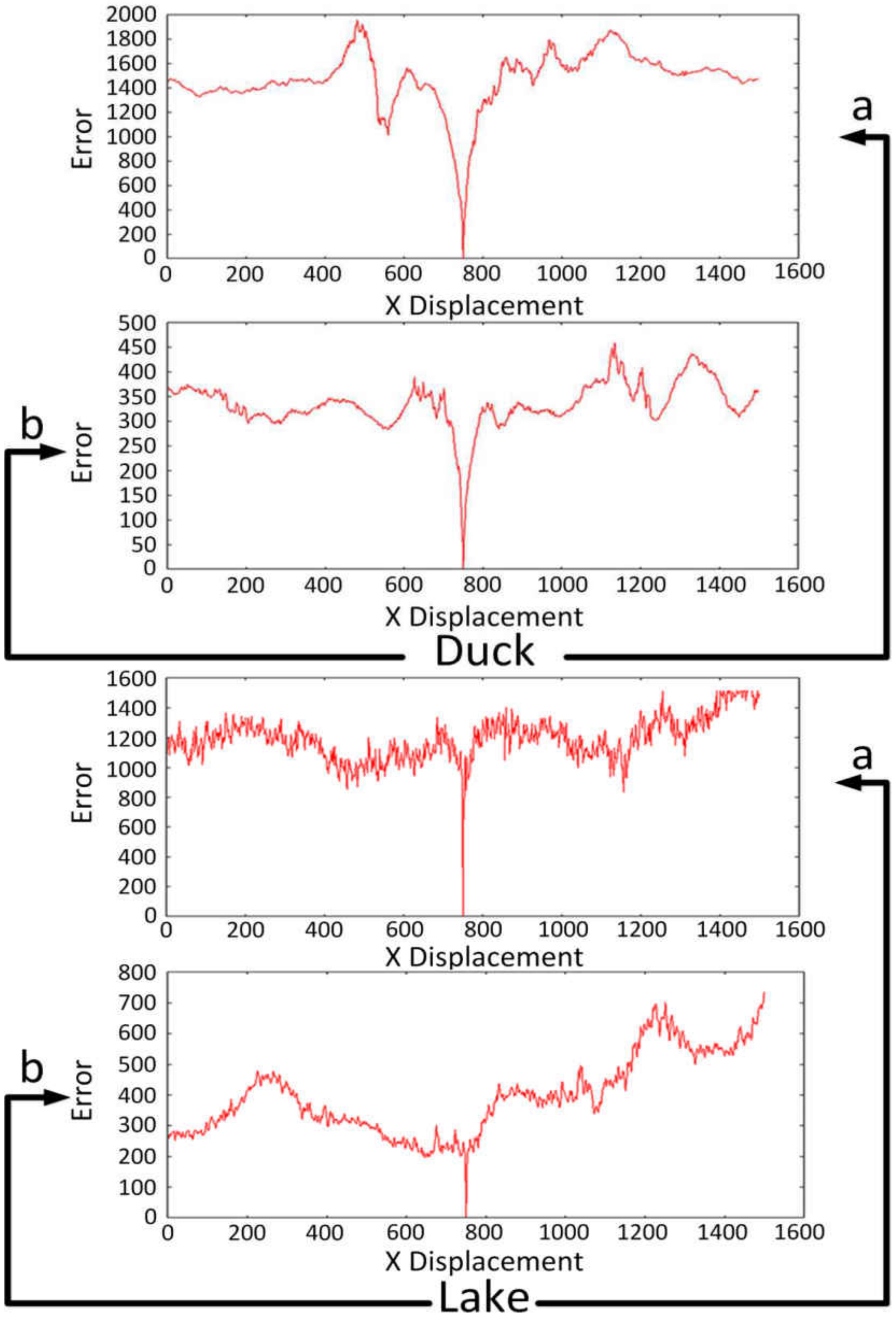


Figure 5.18 - Error curves generated over the best and worst slices of Duck and Lake images

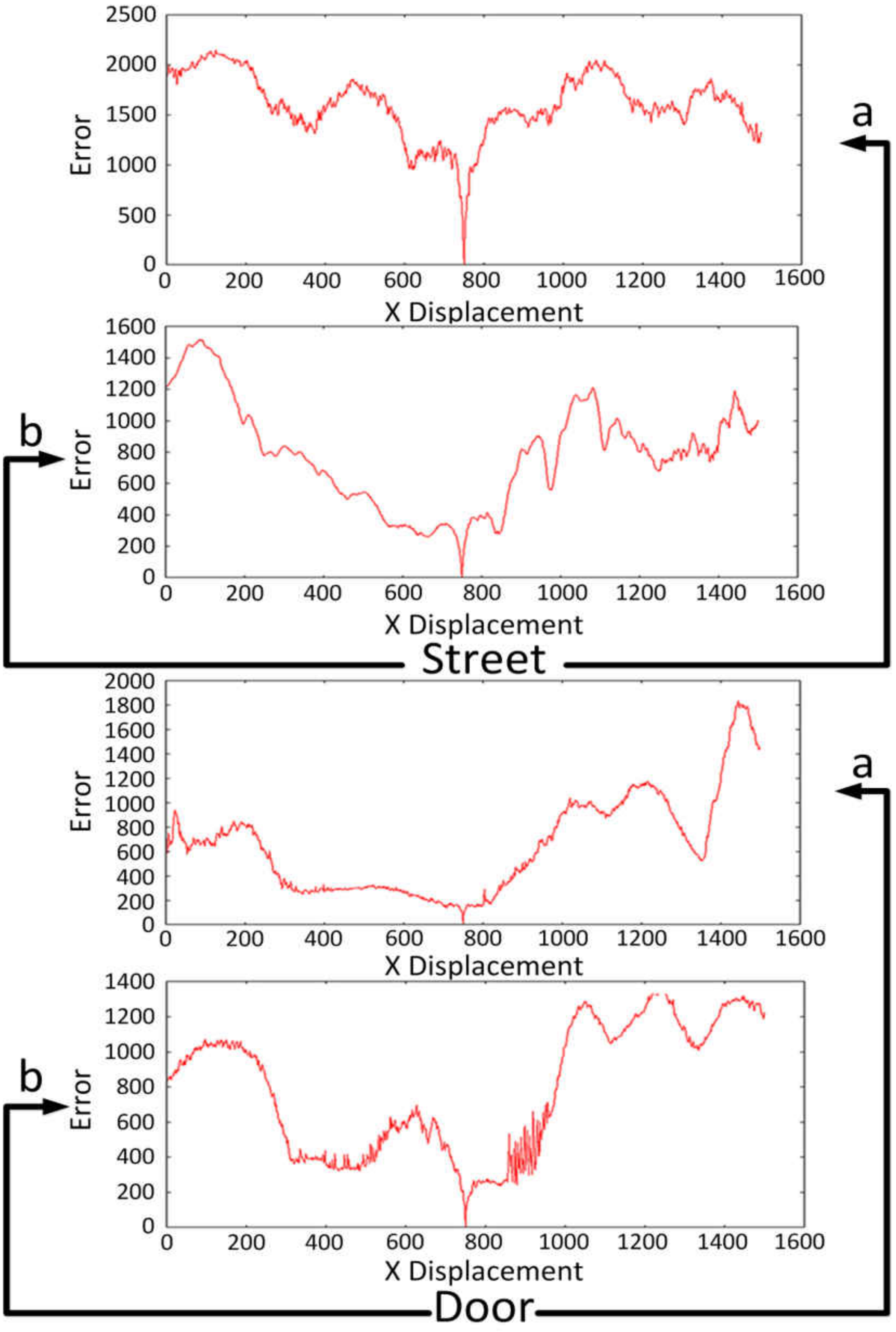


Figure 5.19 - Error curves generated over the best and worst slices of Street and Door images

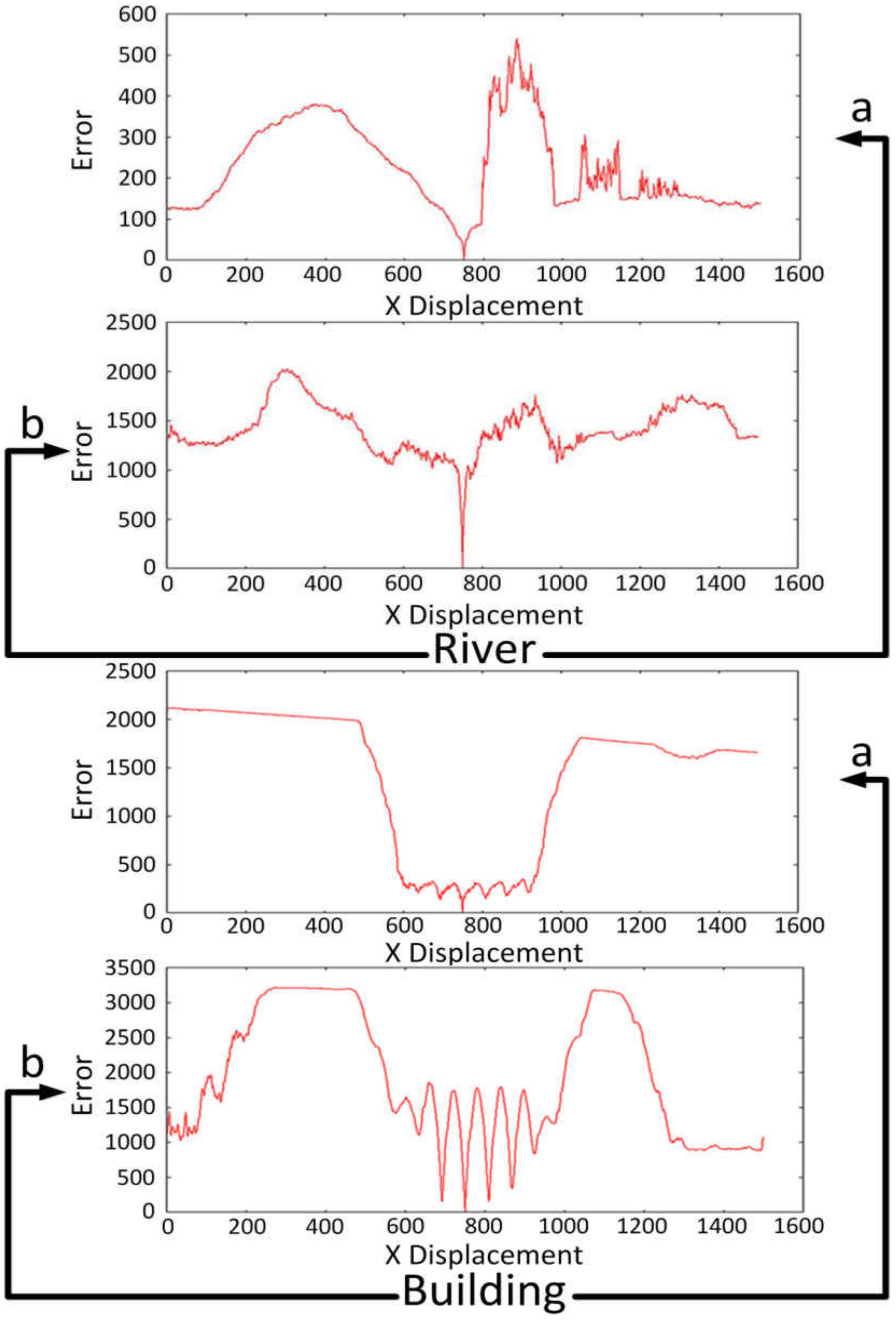


Figure 5.20 - Error curves generated over the best and worst slices of River and Building images

5.6 Edges Distribution

The Error Curves shown in Figures 5.16 to 5.20 are related to how the edges are distributed. We have used the Sobel Edge Detection algorithm [56] (see Chapter 2) with a mask of 3 × 3 pixels to allocate the edges. More edges will cause more gradients on the Error Curves (or more local minimal). Figure 5.21 shows the edge distribution over the ten images. As an example, if we observe the edges distribution over the Garden's image and look back at the Error Curves in Figure 5.17 (Garden), we can see how the edges are clustered around the central feature (red leaved tree). This clustering caused a large descending area on the Error Curve. The sixth slice ("b") seems to have more edges on the right hand side than on the left. This was reflected on the left hand side of the Error Curve, where it shows a highly flattened surface. Having a flat surface is an indication of an area with invariant colouring and with no segmenting edges.

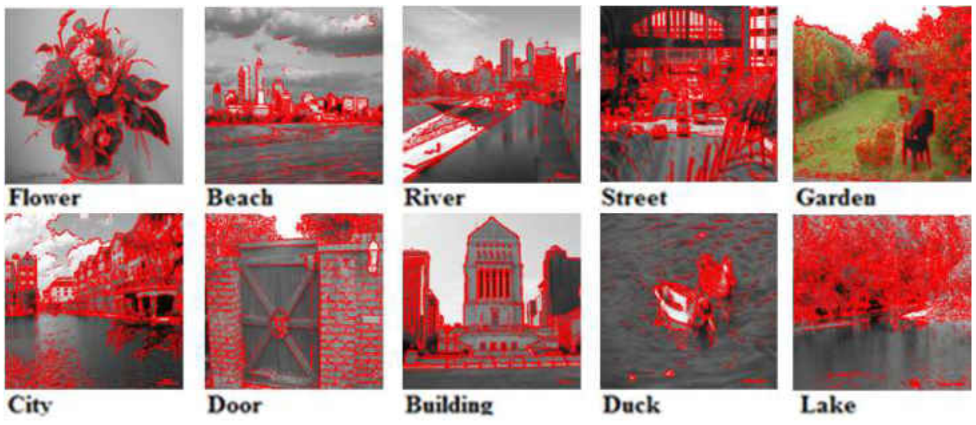


Figure 5.21 – Edges distribution over the 10 images

The Duck, Beach and Flower images have their edges clustered centrally, which reflected on getting the best slices around the central location. For example, the best Duck slice are the fifth and central slices. Slice 6 on the Beach image and 4 on the Flower image were the best slices, where they still counted as central slices. On the other hand, we can observe the lack of edges over the worst slices of the three images. This provides further evidence for the possible relationship between the designed SSP method and the edge distributions. As with the “Beach” example, features like water, buildings and sky are also appearing in the River image and many variant features are clustered in the central position. This clustering reflected on how the error curve was generated over the central slice(s), showing a potential descending curve around the minimal point. However, the generated error curve over the worst slice (River high DCs slice) looks to have a large gradient descending bow around the minimal. However, due to the sharp descending on its right hand side, its potentiality decreases from the controller’s point of view. The sharpness of the gradient may force the controller to be suddenly misled in its direction. The City image shares the same features as in the Beach and River examples (water, buildings and sky) but the difference is in the way those features are distributed. This distribution causes a different edges allocation than what we had in the previous examples. In all three examples (Beach, River and City) edges are clustered around the central slice(s) but in this example (City image), due to having the buildings’ view larger on the right hand side, edges are more assembled to the right than on the left. However, as this assembly is still within the horizontal line, the SSP and the error curves would still show them as potential slice(s).

Every surface has one global minimal but can have several local minima. Most of the retrieved local minimal are a result of running the appearance comparison repetitive regions (i.e. bricks). Figure 5.22 shows how the repetitive regions in the central slice (best slice) affected the ultimate outcome. It shows how the surface suffers from zigzagging points (local points) surrounding the global minimal. However, the global minimal is still identifiable and repositioning to the origin is still possible with one of our Hierarchical Search Algorithms (see Chapter 4). Nevertheless, the risk of getting the Search Algorithm redirected around different local minima still exists, but as explained earlier, the possibility of being recovered on the next run is quite high.

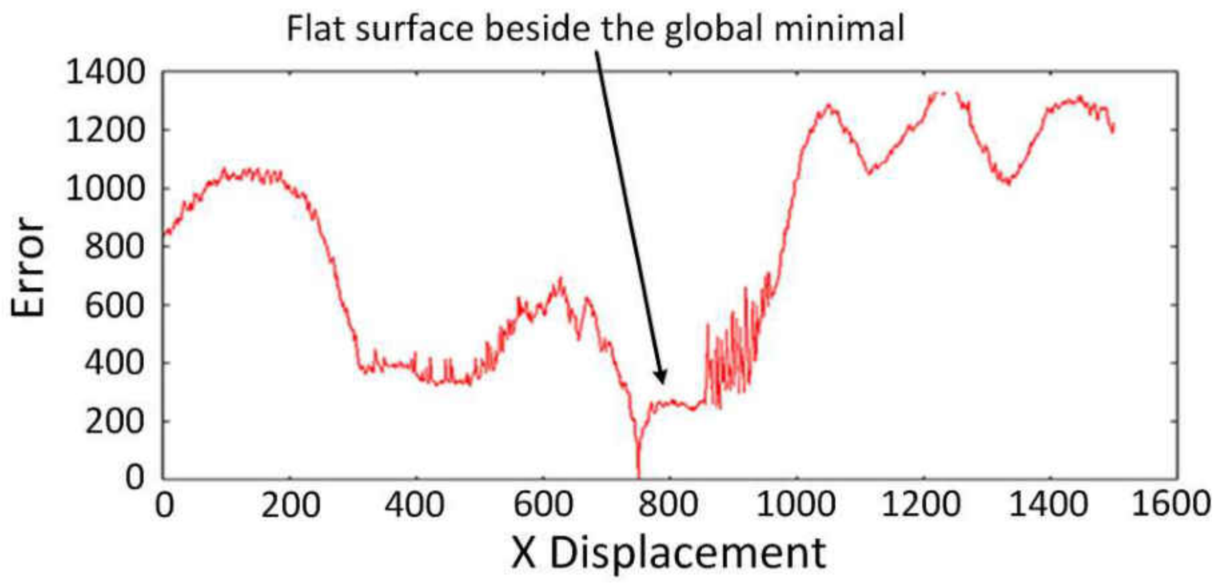


Figure 5.22 – Surface suffers from a large flattened area

The door image suffered highly from region repetitions (bricks and door) which caused a breakdown in relating the SSP to the stabilisation technique (see Figure 5.15). Also, the generated error curves over both the best and worst slices don't appear to have high potentiality where curve "a" (see Figure 5.19 ("Door")) shows a flattened surface on the left which may mislead the navigation in the wrong direction. Curve "b" which was retrieved over the best slice also suffers from a flattened area to the right, which is not as large as the one on slice "a" but still causes issues with defining the global minimal. However, the failure on the Door image was due to having the colour repetition over the central region, meaning the reference patch will encounter many repetitions. Due to having few cars over the Street's best slice, the colour variations on that slice are higher than on some other slices and that may have influenced the wider gradient descending angle which may reduce the possibility of overshoots or losing the global minimal from the controller's point of view. In contrast to the best slice, the worst slice did not include such colour variations (i.e., Cars) therefore, obtaining cars' features is an indication of more variant featuring within the street's view. However, in the printed images, features are static. Yet, in the real world environment, the moving cars are counted as dynamic features and running the SSP over these objects is not a recommended action. Both the Garden and Lake Images are quite similar from the feature repetition point of view, but in contrast to the Garden scenario the Lake image has more colourful clustered leaves which provide more region variety across the slice. However, the generated error curve over the Lake best slice looks to be quite overshooting with many local minimal, but the performance did match the criterion of linking the SSP with the stabilisation performance. More region repetition is shown by the Building image. Figure 5.23 shows the generated error curves over the best and worst slices over the Building image. Both descending areas look quite similar with the difference in their peaks' height (local minima peaks). Therefore, due to having larger peaks around the local minima points, the error curve of the Building best slice looks to have more potential from the stabilisation point of view (see Figure 5.23.b).

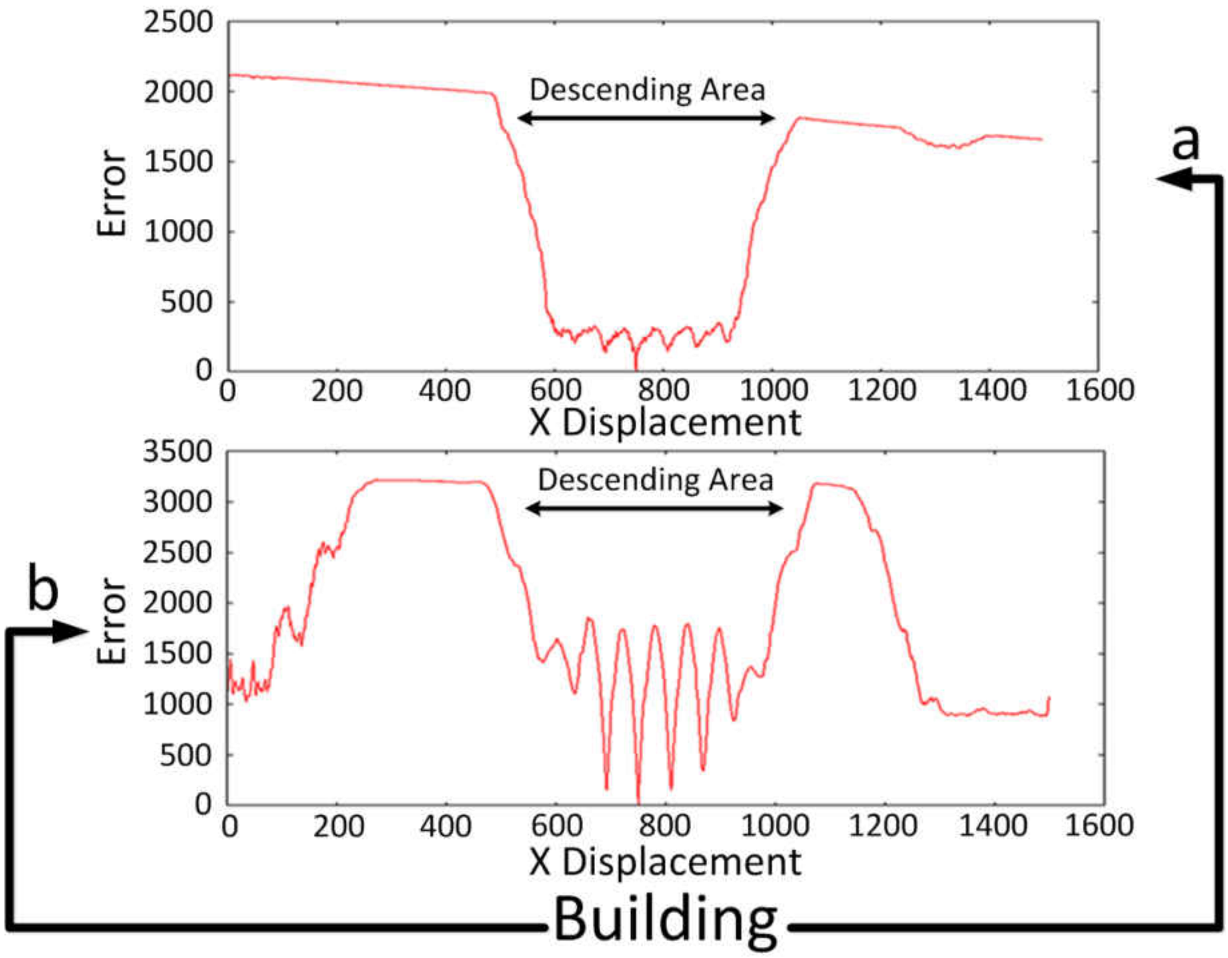


Figure 5.23 – Region similarities leads to few local minimal points

5.7 FFT Analysis

Fast Fourier Transform (FFT) is a widely used signal analysis method [129]. However, it can be used as an alternative technique to analyse the best retrieved pixel displacement graph. To begin with the analysis, we would need to generate a FFT graph for every pixel displacement graph processed over the ten images. Figure 5.24 shows the two retrieved FFT graphs over the best and worst pixel displacement graphs of the Flower image. It shows how the use of the best pixel displacement graph has generated FFT with less signals (or shorter peaks).

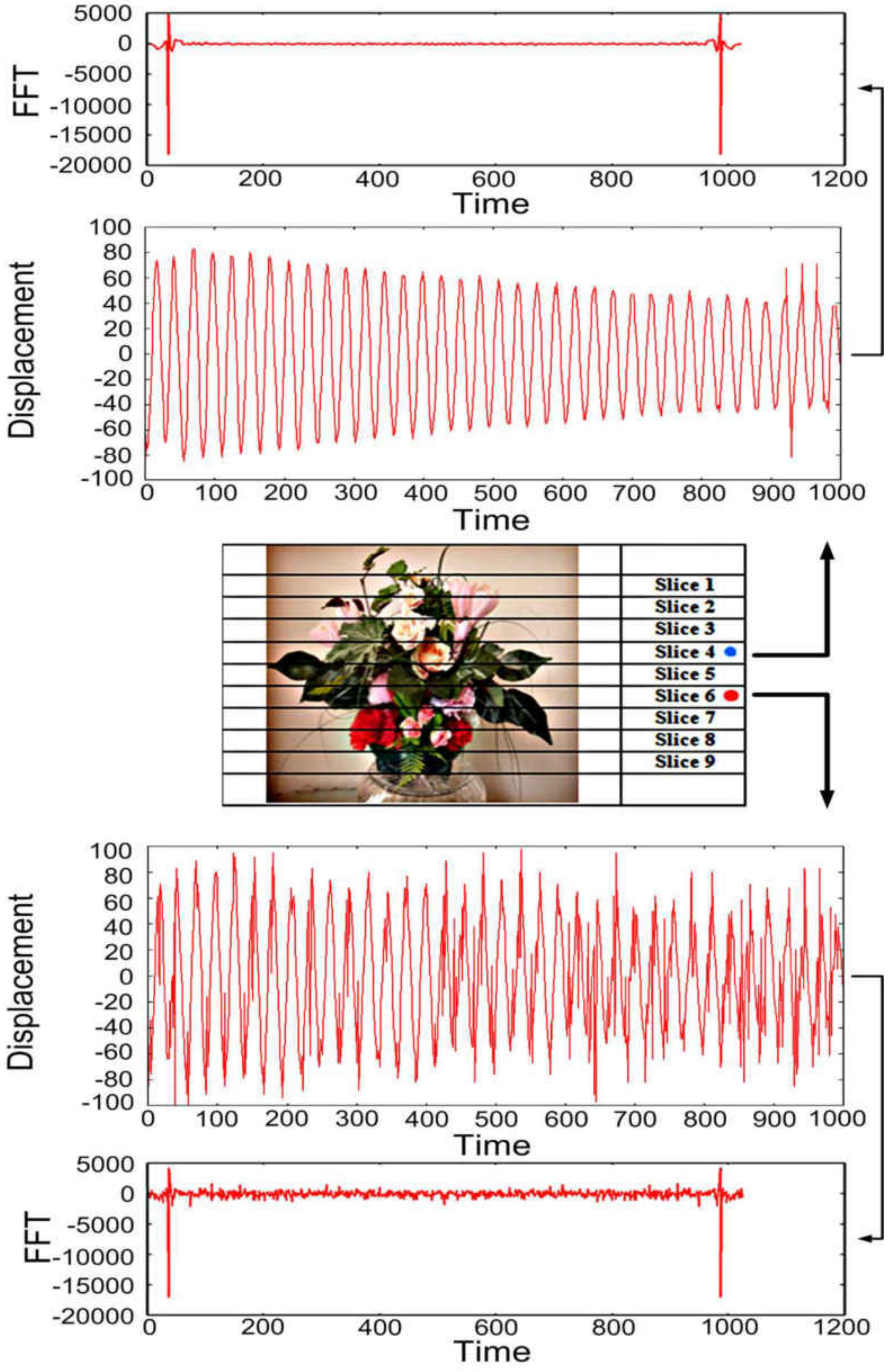


Figure 5.24 - FFT outcomes from the generated displacement graphs over the best and worst Flower image

Here, we get the impression that the generated FFT from the worst pixel displacement graphs would produce higher signal peaks. This introduces the possibility of calculating the total heights of each FFT graph to define the slice with fewer noises. However, the aim is to compare the retrieved DC numbers against the corresponding FFT analysis.

As a consequence of the possible connection between the amount of noises on the signal and the peaks' height, Table 5.2 presents the sum of the peaks' heights for every slice on all ten images besides the corresponding DC values. To make the comparison easier, we divide all retrieved FFT analysis by 1000 and keep them to 1 or 2 decimal places only.

Table 5.2 – DC and FFT analysis numbers for each image's slices

		Images									
		Flower	Beach	River	Street	Garden	City	Door	Building	Duck	Lake
Slice 1	DCs	101	414	606	118	138	285	299	312	328	175
	FFT	42.44	92.42	107.82	38.01	41.98	75.07	65.96	79.77	54.69	77.05
Slice 2	DCs	91	508	427	229	181	156	374	292	303	84
	FFT	28.81	161.47	111.24	72.20	43.94	85.10	124.82	123.58	76.10	31.18
Slice 3	DCs	87	487	204	88	176	108	412	177	257	74
	FFT	28.43	160.87	57.18	25.41	51.19	42.18	150.40	106.71	54.39	29.82
Slice 4	DCs	87	362	95	94	173	82	337	129	124	74
	FFT	28.40	143.17	15.51	24.93	57.37	36.62	164.66	41.91	32.32	30.69
Slice 5	DCs	169	90	109	91	324	109	228	147	76	78
	FFT	46.35	41.22	20.23	26.44	73.92	46.58	55.74	90.24	18.76	30.37
Slice 6	DCs	309	75	87	114	344	205	188	263	102	101
	FFT	122.93	38.40	14.95	34.51	109.72	126.79	69.04	96.05	18.95	32.47
Slice 7	DCs	147	113	159	152	320	452	229	129	117	120
	FFT	36.37	41.56	33.80	44.23	116.43	103.89	85.53	42.80	24.06	36.38
Slice 8	DCs	141	254	143	195	161	419	255	220	251	283
	FFT	49.62	80.86	28.04	41.19	112.84	103.96	82.74	97.00	81.18	81.80
Slice 9	DCs	213	562	291	243	144	548	369	135	436	526
	FFT	37.83	159.64	40.77	45.72	47.41	129.38	160.01	53.27	96.58	152.43

Comparing the FFT results against the retrieved DCs in Table 5.2 shows some matching between the two data sets (FFT vs. DCs). The matching should also be applied to all the images' slices but our major concern here is to identify the lowest FFT and DCs numbers in every image's slices. The goal is to examine if the FFT analysis could still define the pixel displacement graph with the lowest frequencies.

Except for the door and street images, all the lowest FFT outcomes were found to be over the lowest DCs displacement graphs (see Table 5.2). The door image initially failed to match the DCs number against the FFT values. The highest DCs number was found on the third slice, while the greatest FFT value was on the fourth slice. The slice with the lowest DCs number was found over the sixth slice, where the lowest FFT value was referring to the fifth slice. However, the third and fourth slices are both known as slices with high DCs numbers. Therefore, selecting any of the two slices would lead to selecting a useless slice from the stabilisation point of view. On the other side, the lowest FFT was 55.74 over the fifth slice with 228 DCs, while the lowest DCs number was on slice 6 with 188 DCs and 69.04 FFT value. This shows no match between the two slices but it is highly notable that both of these slices have low DCs/FFT values in comparison to the rest of the image's slices' values. Even with this situation, choosing any of the two slices would still give us a useful slice from the low DCs/FFT values point of view.

The collected FFT data for the Street's image is highly likely to be a partial matching than entirely no match. The lowest found FFT was on the fourth slice with the value of 24.93 but the slice with the lowest DCs number (slice 3) has gained a 25.41 FFT value. The difference between the two FFT numbers is quite small. However, due to some possible peaks' overshoot, we may have gained a higher FFT value over the fourth slice. Moreover, the DCs numbers for both the third and fourth slices are considered to be low across all the image's slices. Therefore, choosing any of the two still provides us with a potential slice from the low DCs/FFT value point of view.

Table 5.2 now shows that those displacement graphs with a higher number of peaks are also seen to be graphs with many noisy signals from the FFT point of view, therefore, we can use this argument to scientifically back the DC technique even further.

However, as Table 5.2 shows, the majority of our FFT analysis fully matched with the DC measure using the pixel displacement graphs. Though the use of FFT may well be used as a substitute for the DC measurement, the question is which analysis looks more accurate within our project scope.

Figure 5.25 shows two graphs, where the right hand side one (red graph) refers to a scenario that has a constant swing with invariant changes to the image space coverage, but the left hand side displacement graph (blue graph) presents a state where the platform's swing started from covering a large area, down to small area, then back to coverage of a large area.

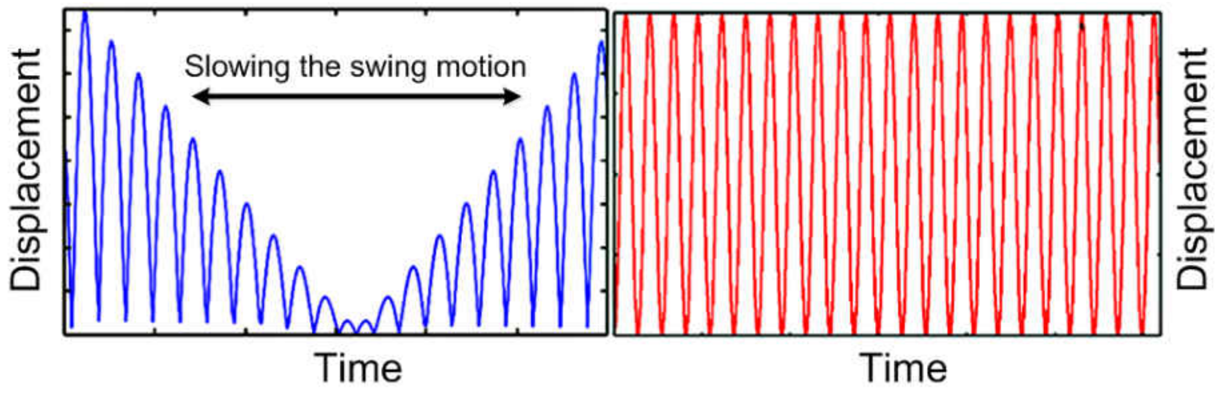


Figure 5.25 – Two displacement graphs with equal DC value

Figure 5.25 shows that both graphs have equal DC numbers but they differ in how the platform's pendulum was performed. Though from the FFT analysis point of view these two graphs may differ, they both have an equal DC number. However, due to not having any overshoots, we can clarify that both pixel displacement graphs were retrieved precisely. Therefore, if the FFT process shows a different analysis on both graphs, this will be an indication that the FFT process has failed to manage the graph analysis. As a consequence of this the DC measurement is a safer way to analyse our pixel displacement graphs.

5.8 Conclusion

From the above analysis, we clarify the possible effect of the edges' cluster on the ultimate SSP/Stabilisation outcome. However, this edges' clustering is also required to have large colour variations in the captured region. For instance, we found a large number of edges on the best slice of the Lake image but due to the colour variations, it introduced a great region variation across the slice. If this edges clustering had low colour variations (i.e. a tree with leaves of a single colour), it may introduce large region similarities over the monitored slice (i.e. the Garden's worst slice (tree leaves)). Figure 5.26 shows an example of two slices, where the bottom slice suffers from colour variations and introduced high region similarities.

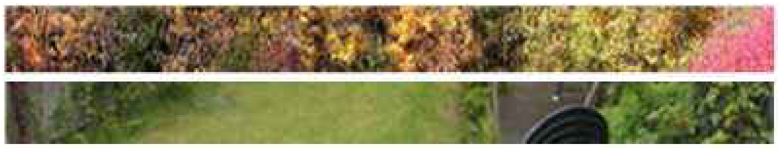


Figure 5.26 - Top is the best Lake slice and bottom is the worst Garden slice

This shows how the edges distribution can become an advantage or disadvantage depending on the region colour variations, therefore, from our tracking point of view, the region variations is more important than how the edges are distributed. Nevertheless, the size and position of both regions need to be considered, while the region's size should not be larger than the captured patch. Having the targeted region larger than the patch size will introduce local minima around the global minima and results in misguiding the tracking process. Moreover, referring back to Design Methodologies chapter (Chapter 4), it is preferable for the position of this region to be as centralised as possible to allow the most promising patch iteration steps on both horizontal sides.

Beside the use of the DC (Direction Changes) as a major measure, we have also used the Fast Fourier Transform (FFT) analysis to support the usability of our SSP outcome with a well-known used analysis function and also support.

As the applied experiments' results were quite promising, the next chapter will concentrate on applying the implemented Slice Selection Process (SSP) and the hierarchical search algorithm in real world environments.

Chapter 6

Real World Experiments

6.1 Preface

In the real world experiments, we aim to repeat the testing operation that was achieved in the laboratory in real world environments and therefore, we will assess the credibility of our laboratory experiments a bit further. The main concern is to have sufficiently variable illuminations and wind speed to force the platform to swing or capture images differently.

The purpose of these experiments is to see how the designed system performs in real world scenarios. As is well known, there are many issues that are faced when moving from the laboratory to real world projects. Many of the world's projects were designed to work in the laboratory but they never succeeded in the outside world. In our laboratory experiments, many environmental conditions (i.e. lighting) were static and we also had good control over the platform's ground distance, which allowed us to set the camera's lens in the best possible way. The real world scenario differs according to changes in the lighting level and winding conditions etc. Therefore, the lens' setting requires manual changes from time to time. However, wind is counted as a major issue as its speed variation will reflect over our swing acts. We did not encounter this issue in the laboratory environment where we used to push/pull the platform by a pre-specified measured distance for a pre-specified period. Although, we are aware of the fact that in the laboratory experiments our first push gave the swing the highest power and that the power decreases as the time increases, which would cause the platform to cover less image distance. Therefore, in contrast to the laboratory where the swing speed can be pre-specified, the real world environment can force the swing to obtain sudden changes and act randomly throughout the timing period.

In our laboratory work, if the platform's camera loses the origin, it would be our influence that will help the platform to get redirected back. However, in the real world scenario, the winding condition is the only influential command that can help the tracking direction to either lose or fix its wrong tracking path. Lighting is the next major issue that needs to be dealt with as much as possible. As we are aware, the lighting can cause the object's colour to differ according to the lighting level. As explained in the literature review chapter (see Chapter 2), the RGB colour component does not take the illumination into account. This can cause the possibility of the colour of one or few objects changing during the tracking period, which may redirect the performance from best to worst. In the laboratory, we had a constant lighting condition which eliminated the risk of the illumination changes. This lighting level constancy enabled us to adjust the camera lens and set the gain and exposure variables only once. However, due to the constant change in the sun's position, this lighting constancy will become impossible and we may need to apply those settings more than once. Moreover, some external lighting can have a significant affect on the surrounding lights; for instance, in Figure 6.1 (Outdoor) an electrical light is located on the top of the wall (below the windows) and goes on and off frequently. The effect of this on and off is greater when the sun light becomes weaker, which is highly reflected over the entire stabilisation procedure.

However, having the electric light on the top and pointing towards the targeted area made the colour changing highly reflected by the change of lighting conditions. As mentioned earlier, we changed the gain and exposure variables more than once in the real world experiments, but defining the best setting was achieved by running the camera's program (uEye) with live image broadcasting and manually changing those settings to look visually clear from any possible blurriness. However, human visual systems are quite good at judging this blurriness level. Our aim is to supervise the entire experiment during their operation. This supervision allows us to visually judge the system performance and have our own influence whenever it is needed (i.e. redirection to a wrong location), but we have chosen three different locations to apply our testing on. The first was inside the garage to give variable lighting levels without the windy conditions.

The second experiment is in an outside area where both light and wind are variant. Finally, the third is the sea experiment where we used a boat to apply the testing. Figure 6.1 shows the three images of the three experimental environments, with the arrows indicating the direction of the platform's motion in each scenario.

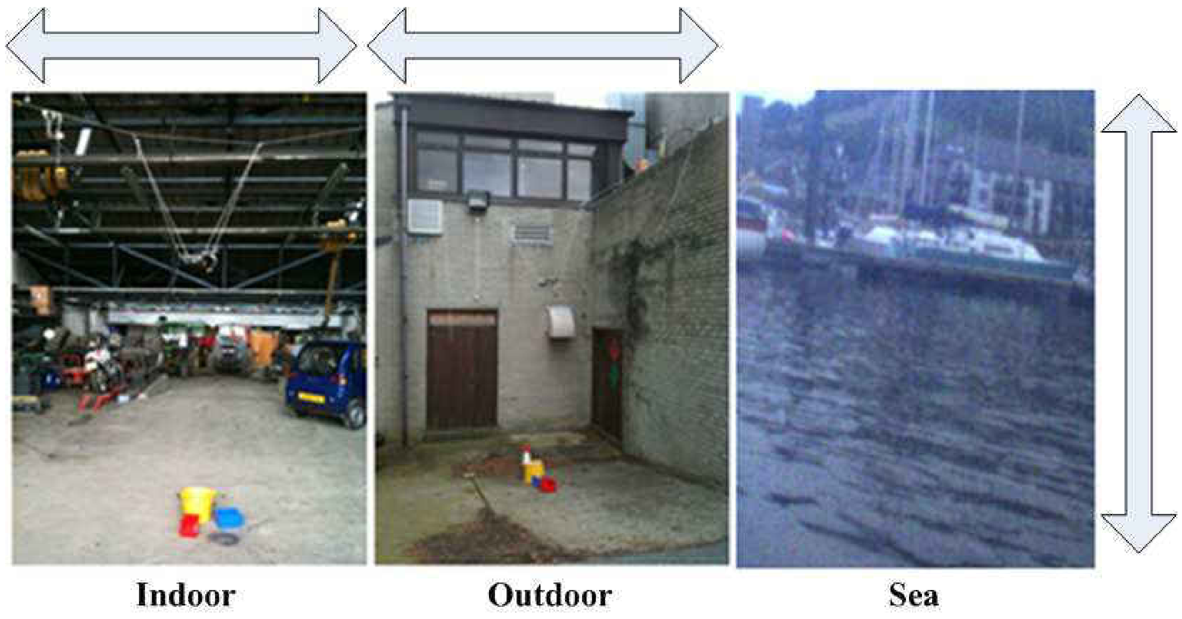


Figure 6.1 – Indoor, Outdoor and Sea environments

The three test locations were selected for ease of deployment and the possible variation in both light and wind conditions. The Indoor image in Figure 6.1 shows relatively dim lighting and is slightly affected by the passage of clouds; the loading area (middle image) is exposed to wind and the open sky above it so is much brighter and affected to a large degree by the clouds' passage and the sun. Ultimately, the sea environment was used to test the reliability of the system performance in a different scenario.

In sea experiments, the central slice would mainly hold the majority of the region variations and therefore we would exclude the use of SSP and employ the most central image region as a reference to stabilise the vertical axis. However, as we are excluding the SSP method, we will start our real world experiments on the sea environment first to test the credibility of our searching algorithms when used in the outside world.

6.2 Sea Experiments

6.2.1 The Aim

As the aim of this research is to address the problems that are inherent in the visual stabilisation on those platforms which suffer from cyclic motions, we need to examine this platform on different scenarios. As a consequence of this, we have decided to affix this platform on the boat to see how it would resolve the stabilisation issues caused by the boat's cyclic motions. However, the two scenarios differ in the way the searching direction operates (see Figure 6.1 (Arrows)), whereby in the Kite scenario, the searching algorithm operated over the horizontal axis, whereas in the sea experiment, the method was designed to operate over the vertical axis.

The experiment here aims to show that the technique developed for stabilisation of the kite platform will function effectively for platforms which suffer from cycle motion.

However, the region variations in the boat scenario are meant to be more unpredictable than in the Kite scenario. This unpredictability is due to that fact that when the boat moves we encounter a view change frequently. Moreover, as explained earlier, the colour constancy is more possible in the sea experiment, where the water view is on the bottom slices, the top slices are dominated by the sky/clouds view and the middle slices benefit from the region variations. However, unlike the kite's scenario, the region/colour constancy over the sea experiments is an advantage. This advantage is due to the fact that having the region variations in the middle slices encourages us to exclude the use of SSP. We can still encounter small or large boat movements, but as we try to keep the boat more stable we should still keep the view or part of the original view existent within the frames. Nevertheless, if we intend to let the boat move along, then stabilisation is required by running the appearance comparison between every two consecutive images rather than the consecutive frames with the stored reference.

6.2.2 Reasons for choosing boat

Boats suffer from continuous tilt angle changes which introduces a continuous cyclic motion. The platform is meant to resolve the stabilisation issues caused by this cyclic motion.

6.2.3 Process Plan

The boat was equipped with a fixed platform to one of the Boat's sides connected to the PC with a 240V charger. We commence by choosing a few views (at least 5) and allow the camera to point horizontally towards those chosen views on each one of the experiments. To increase the repeatability of the results, each experiment was applied five times. Views were selected according to their region and colour variances, which then provide an overview of how well this method can operate within the sea environment.

6.2.4 Issues & Difficulties

Power supply and the weather conditions were the greatest issues that we encountered during our sea experiments. The charger had to be charged after 2-3 hours of usage and charging time was also 2-3 hours. In most cases we charged the battery overnight and performed the experiment the day after. More time wastage was incurred during the boat parking; uninstalling the platform, packing all the equipment, driving back to the lab then recharging the battery and returning to the same place to repeat the same steps to install the system back into the boat.

6.2.5 Applying the Process

Due to being able to use the SSP technique, we have designed the code to capture 10 images over the experimental period. However, we will align the images jointly and draw a line from the first to the tenth to determine the platform's stability over the vertical axis. This definition is achieved by identifying a common region across the whole images' sequence (if applicable). We then calculate the pixel displacement of each labelled region from the drawn line to determine the margin of their position variance across the sequence.

Below shows five experiments from five different views, where each experiment was repeated at least 5 times. We demonstrated all the gained results using tables but for each experiment we have selected two out of the five tests to present their ten captured frames visually.

6.2.5.1 First Experiment

Figure 6.2 shows the entire view of which the first experiment is to be applied on. The image shows the sky and water has dominated most of the shown space.

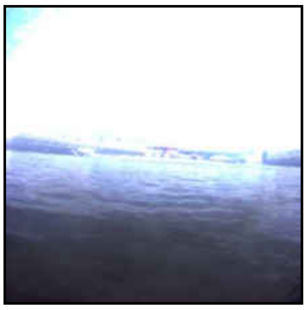


Figure 6.2 – Area view for the first experiment

We can visually judge that the central slice is highly distinctive and its tracking should be very reliable from the searching algorithm point of view. Figure 6.3 shows the 10 captured frames for one of the applied tests. These images were captured based on a fixed timing period from the attached camera to the boat's side. As explained earlier, we start by selecting the most common region within the whole frames' sequence (or the majority). We try to show if the selected common regions are still located around the same position in all frames throughout the sequence. The common region position in all frames will be shown by their pixel positioning, therefore, a good stabilisation performance is expected to be within the same area from the pixel positioning point of view. In Figure 6.4, we have selected the boat's bottom part as a common region. We have drawn a line below the boat's base over the frames' sequence. We also indicated a common region (below the boat) with a black dot over the common area in all frames. Observing the pixel displacement variance of the dot from the drawn line will specify how well the camera was stabilised over this view. However, due to the fact that every frame is unique amongst all captured frames from the blurriness and the lightness point of view, it then makes every frame differ from the RGB colour space from all captured frames within the sequence. These differences make the process of common region selection using software based power more difficult, therefore, all our region selection techniques are made using eye control only, but to determine the common region's locations we use the Photoshop program to specify the exact pixel positioning of the indicated common regions.

The indicated dot on the frames' sequence (see Figure 6.4) shows a little pixel displacement variance from the first to the fourth frame. However, the common region disappears on the fifth frame, appears back in view on the sixth frame but disappears again on the seventh frame. Ultimately, it recovers well on the very last two frames. Frequently loss and recover are caused by either the environmental conditions (i.e. wind) or the water's current. The water's current causes significant changes in the boat's tilt angle but in most scenarios it is our influence that changes the title angle by the push and pull of the boat's sides to cause more non-stabilised conditions. This gives our stabilisation technique more credibility in much worse windy conditions. This push and pull caused the camera to completely lose the view (i.e. Figure 6.3 (Frame 8)), but the searching algorithm was able to recover the stabilisation quite well.

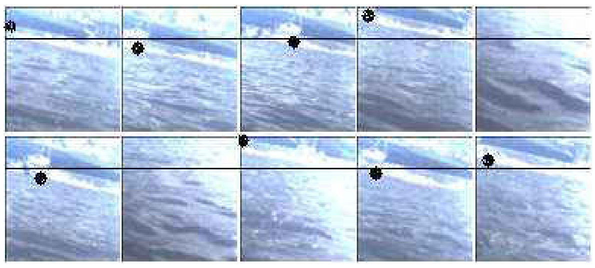


Figure 6.3 - Stabilisation process in the first test of the first experiment

Figure 6.4 shows another test in the same experiment. As in Figure 6.4, dots were also located below the boat's surface, but in this test the images' brightness looks higher. However, the only loss on track was in the fifth frame but this could be recovered in consecutive frames.

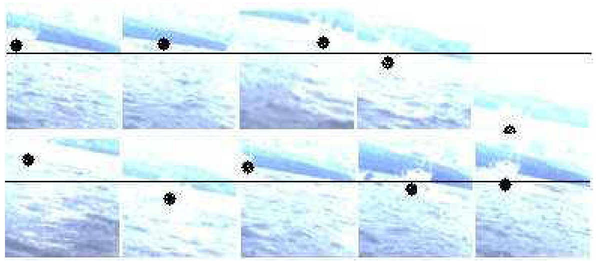


Figure 6.4 - Stabilisation process in the second test of the first experiment

Stabilisation succeeded in both scenarios (Figures 6.4 & 6.5) and in addition to the weather conditions, it was our influence that gave the boat a more non-stabilised condition. Therefore, the region variance on the central slices (Figure 6.3) has shown to be more stabilised from the tracking point of view.

Table 6.1 demonstrates how the pixel displacements gained from the five repeated tests over the first experiment. The standard deviation of each test and their average is also presented. The standard deviation would provide us with the range of variations in our pixel displacement data. However, we have gained negative values over the second, third and fourth tests where they represent the camera's overshoot during the tracking period.

Table 6.1 – Pixel displacement values of the first sea experiment

Experiment 1	Test 1	Test 2	Test 3	Test 4	Test 5
Pixel Displacement	30	7	4	23	28
	11	5	15	11	33
	27	-10	48	-15	42
	5	10	59	12	45
	59	-15	26	7	30
	22	2	-5	10	56
	12	10	14	2	46
	28	13	15	21	48
	15	19	27	-5	56
	18	15	21	3	52
Standard Deviation	15.14	10.77	19.14	11.40	10.26
Average	13.34				

However, as the standard deviation is meant to show the variance of the gained data, to retrieve the variance of the retrieved pixel positions to the target, the standard deviation is an ideal method to use. Moreover, from the camera's stabilisation point of view, having negative pixel positions should increase the variance to a higher value but using different methods such as the Median could decrease this variance and make it look like a more stabilised condition.

Table 6.2 shows an example of how the standard deviation was more realistic than the Median value. Table shows a non-stabilised scenario where we move from position -15 to 15 in every step. The standard deviation is shown to be high, whereas the Median was calculated to be 0, therefore, to observe how well the platform was stabilised the standard deviation is a more realistic outcome.

Table 6.2

Position	-15
Position	15
Position	-15
Position	15
Position	-15
Position	15
Position	-15
Position	15
Standard Deviation	16.035
Median	0

6.2.5.2 *Second Experiment*

Figure 6.5 shows the view of the second sea experiment. As with the first experiment, to increase the repeatability, the test was applied five times. However, comparing this view with the view on the first experiment, the region variances are less but their colourings are more saturated. The saturation in the central slice is caused mostly by having a large red boat but the sky and water dominations in the top and bottom slices are highly obvious. This distinction between the top and bottom slices is caused by the central slice and can significantly help to keep the camera stabilised over the middle view.

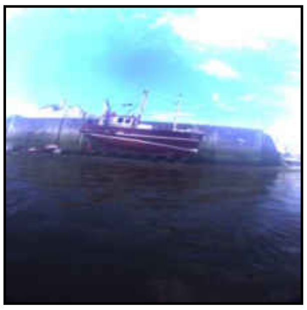


Figure 6.5 - Area view for the second experiment

Figure 6.6 shows the 10 frame sequence captured during the tracking period of the second experiment. Figure 6.6 shows the indicated common regions are quite close to the drawn line throughout the whole frames sequence. This is a good indication that the camera had some sort of stabilisation over the vertical axis. In contrast to the first experiment, the lighting strength looks lower. This light weakness could be due to the sun's position as this experiment is almost 180 degrees opposite the view of the first experiment. Nevertheless, we are still able to change the lens or the camera setting to enhance the handling of the lighting condition but we decided to allow the platform to encounter all possibilities with a similar setup. However, the capturing of a good image is highly dependent upon the lighting circumstances; meaning, low lighting level increases the darkness and causes some features to disappear from the visual point of view. On the other hand, the high brightness decreases the saturation and increases the similarities amongst the features.

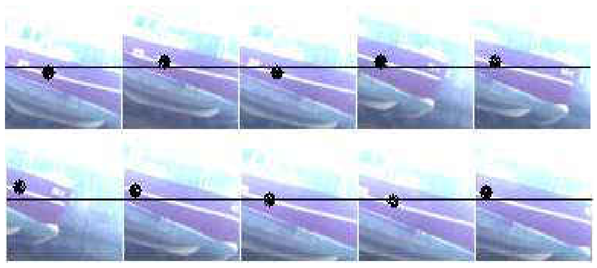


Figure 6.6 - Stabilisation process in the first test of the second experiment

The next test in the same experiment shows a slight increase in the environmental lighting. This light boost still kept the red as a highly saturated colour within the scene. By observing the indicated dots on the images' sequence, we can notify how well the stabilisation was achieved. However, our push and pull gave a non-stabilised condition which can be noticed from the second to the sixth frame but the system could recover back on subsequent frames. The stabilisation mainly occurred over the first, seventh, eighth, ninth and tenth frames in Figure 6.7. On the other hand, due to having a large redness across the central slice, when the camera is shifted to the right or left, it will cause a change in the redness amount/position within the view which would reflect over the stabilisation performance dramatically.

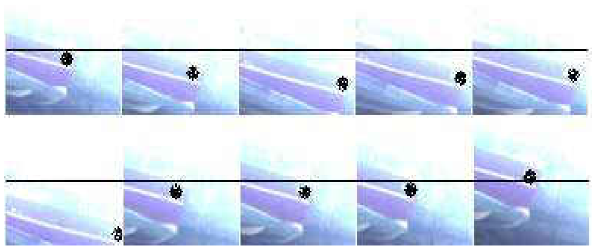


Figure 6.7 - Stabilisation process in the second test of the second experiment

Table 6.3 demonstrates the gained pixel displacements for all five tests over the second experiment. The gained standard deviations of the second experiment looks to be lower when compared against the first experiment numbers. However, this reduction indicates a better stabilisation in the second experiment.

Table 6.3 - Pixel displacement values of the second sea experiment

Experiment 2	**Test 1**	**Test 2**	**Test 3**	**Test 4**	**Test 5**
Pixel Displacement	39	48	55	37	48
	40	51	56	37	43
	42	50	55	38	42
	35	45	50	59	44
	36	38	46	45	42
	35	43	49	47	43
	39	43	49	49	40
	42	51	43	44	42
	33	45	49	38	40
	35	51	54	29	41
Standard Deviation	3.20	4.42	4.42	8.34	2.32
Average	4.51				

As Figure 6.8 shows, the domination of the sky and water's view is less in experiment 2 than in experiment 1. However, as image 1 (Figure 6.8) demonstrates, the first experiment is closer to usual real sea scenarios than in the second experiment where the boat's body was the main target to be tracked. This shows how the domination of one colour can force the system to perform differently. Image 1 on Figure 6.8 is much closer to being a horizon tracker, whereas the second experiment seems to be more of an object tracker (or boat tracker) in the sea environment.

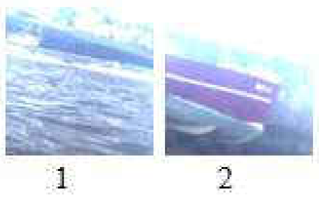


Figure 6.8 – Frames from the first and second experiments

As a consequence of the comparison between the first and second experiments' outcomes, where the camera in the second experiment was more stabilised, here we have learnt that due to having the sky and water as the primary views in sea experiments, having high saturated colour (i.e. red) on the horizon level would help to create a more stabilised scenario.

6.2.5.3 Third Experiment

Figure 6.9 shows the view of the third sea experiment. As in first experiment, the sky and water have dominated the top and bottom slices. However, the central slice benefits from the region/colour variation which provides a great distinction between the top and bottom slices. A few boats, a white building and a mountain in the background were behind the region variation of the central slice.

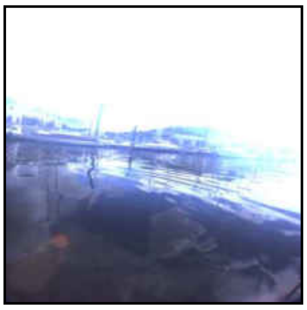


Figure 6.9 - Area view for the third experiment

The frames' sequence in Figure 6.10 demonstrates one of the five tests over the third experiment. The black dots are located below the boat's surface and it indicates how well the stabilisation technique was achieved.

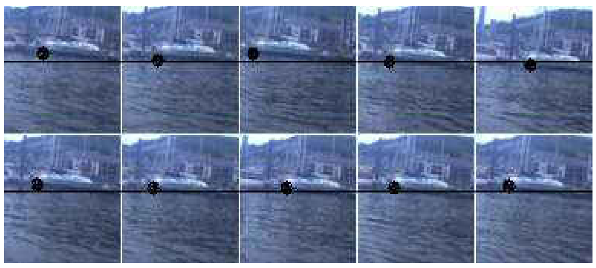


Figure 6.10 - Stabilisation process in the first test of the third experiment

In contrast to the second experiment, the third experiment tracked a larger view with wider region variations. However, as Figure 6.11 shows, the camera shift in the second experiment caused a more non-stabilised condition in comparison to the third experiment scenario. In the third experiment, the camera was shifted to the right but the central slice was still the same distinction level between the top and bottom slices. In the second experiment, the boat surface dominated a large part of the view and therefore the camera's shift caused the system to act more as an object tracker than a stabilisation process.

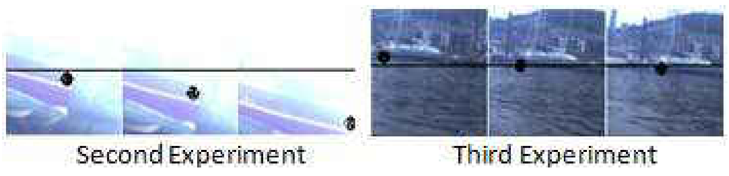


Figure 6.11 - Frames' sequences from the second and third experiments

In addition to what we learnt from the comparison between the first and second experiments, we have also learnt that the more object variations we have, the more independent that we are from stabilising over a single object.

Figure 6.12 shows another frames' sequence of a test over the third experiment. We push and pull of the boat's sides to make further instability. The indicated black dot completely disappeared from the second, third and fourth frames but came back to the scene in consecutive frames. This demonstrates the system's ability to recover when it encounters a large instability condition.

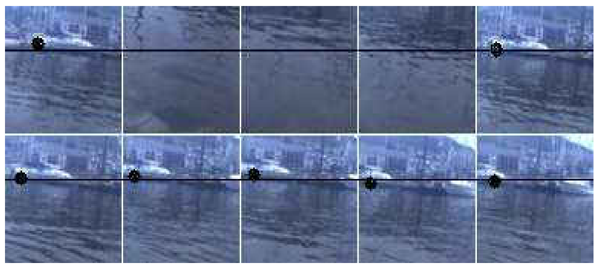


Figure 6.12 - Stabilisation process in the second test of the third experiment

Table 6.4 presents the retrieved pixel displacements, standard deviations and their average value of the applied tests over the third experiment. The displayed negative numbers are due to the physical camera's overshoots, which mainly appeared during the fourth and fifth tests. However, the gained average value is less than the gained average on the first experiment but it is larger than the retrieved value over the second experiment. This experiment was applied when the weather was too cloudy and there was less sun. This caused the whole environmental view to appear darker and decreased the saturation of some of the saturated colours we had in the observed view. This darkness caused significant colour similarity between the hill in the background and the water's view (see Figure 6.12) which caused the camera to point to a different view. Therefore, we have learnt that dependency on the colour saturation highly decreases when the sun light decreases and this may cause an instability scenario.

Table 6.4 - Pixel displacement values of the third sea experiment

Experiment 3	Test 1	Test 2	Test 3	Test 4	Test 5
Pixel Displacement	16	21	20	14	22
	14	22	19	-45	19
	17	22	17	-32	19
	18	19	18	-43	21
	15	24	18	18	22
	17	21	18	-32	-32
	16	17	19	17	-35
	14	19	19	17	-20
	18	18	17	14	20
	17	21	17	18	21
Standard Deviation	1.47	2.11	1.03	28.37	24.25
Average	11.45				

6.2.5.4 Fourth Experiment

Figure 6.13 shows the fourth sea experiment's view. In this view, the colour variance and the way they are distributed is quite similar to the first and third scenario. Water, sky, building, bridge, boat and mountain are the dominating features in Figure 6.13.

Figure 6.13 - Area view for the fourth experiment

Figure 6.14 shows the frames' sequence of the applied test on the fourth experiment. Water, boat, bridge and buildings' view in the background are the main features shown on this frames' sequence. The black dots were located on the front part of the boat's body. As with the first and third experiment, this test has shown a great stabilisation performance when the camera is left or shifted right. However, some instability occured over the eighth frame but the system could recover itself in the consecutive frames. The boat and bridge were the main distinctive objects which caused the tracking algorithm to perform well over the central slice(s). Also, the bridge is a highly textured feature which can have a great influence over the tracking process. Having distinct objects (boat and bridge) crossing the central slice gives high reliability over the tracking procedure, even when the left or right shift occurs.

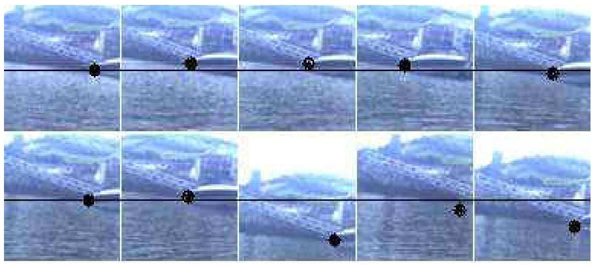


Figure 6.14 - Stabilisation process in the first test of the fourth experiment

Figure 6.15 shows a frames' sequence of another test over the same experiment (experiment 4). Boat, buildings and hills in the background were the main features within the scene. Also, the bridge in Figure 6.14 dominated the majority of the central slice button, in Figure 6.15 the boats and buildings are less visible and the bridge was the dominating feature. However, having fewer visible features was caused by having a larger distance between those features and our camera. This distance enlargement was decreased by the water's current, which pushed our boat more towards the features. Therefore, the features in the last frames seem to be more visible. The camera remained stable but we did encounter some instability if the boat was getting closer to the targets, therefore we have learnt that during periods of darkness, the more distance we have from the target, the more possible instability we may encounter.

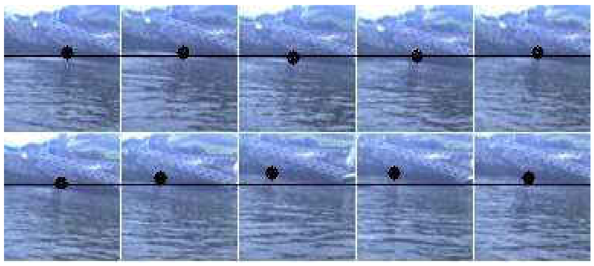


Figure 6.15 - Stabilisation process in the second test of the fourth experiment

Table 6.5 shows the pixel displacements, standard deviations and their average values for the five tests of the fourth experiment. Our only overshoots in this experiment appeared to be over Test 4, which increased the standard deviation to 18.50. However, the gained average is now 7.19, which is less than the first and third experiments but higher than the second experiment's average.

Table 6.5 - Pixel displacement values of the fourth sea experiment

Experiment 4	**Test 1**	**Test 2**	**Test 3**	**Test 4**	**Test 5**
Pixel Displacement	20	18	12	-32	19
	21	39	11	11	11
	21	16	15	11	21
	21	18	17	14	23
	22	20	17	10	21
	24	16	18	13	17
	23	16	17	17	16
	24	14	15	13	25
	24	17	15	12	32
	20	29	12	-30	21
Standard Deviation	1.63	7.76	2.46	18.50	5.62
Average	7.19				

Nevertheless, as Table 6.6 shows, if we exclude the standard deviation of the fourth Test, the new average becomes 2.84 and that is less than any gained average of the previous experiments. This demonstrates how overshoots can have a significant affect on the ultimate outcome.

Table 6.6 – Excluding one standard deviation value from the fourth experiment and calculate the new average

Standard Deviation	1.63	7.76	2.46	5.62
Average	2.84			

6.2.5.5 Fifth Experiment

Figure 6.16 shows the view of the fifth sea experiment. As in most of the previous examples, the top and bottom slices are dominated by the sky and water's view. Having the white building and stones' wall on the right hand side gives a large region repetition over the central slice; however, it also helps to create a highly distinct region between the top and bottom slices.

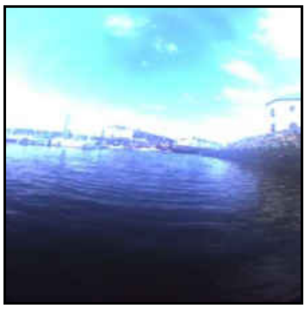


Figure 6.16 - Area view for the fifth experiment

Figure 6.17 shows the frames' sequence of the stabilisation process over the fifth experiment. A few boats and buildings in the background are the main features within this frames' sequence. However, the boats and buildings are white colour features which can introduce high colour repetition over the central slice of the frames' sequence. This may not differ from the human's visual system point of view but it is a highly considerable point from the image processing side. This means that the mismatch during the tracking period is still possible but the region below the buildings and on top of the white boat has a great colour variance, which can make a difference and mean the stabilisation is successful. The black dot on the frames' sequence is located below the white boat which demonstrates how well stabilisation was achieved. Moreover, the water's view can help to distinguish the colour variance of the central region and facilitate the tracker to define more precise pixel displacement.

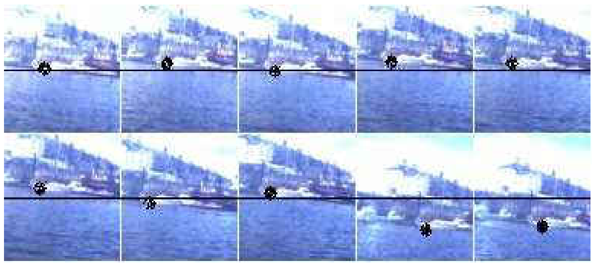


Figure 6.17 - Stabilisation process in the first test of the fifth experiment

Figure 6.18 shows the frames' sequence of another applied test over the fifth experiment, but a higher lighting level has illuminated the regions. However, as mentioned earlier, higher illumination reflects in decreasing the colour saturation and increases the similarities between some represented colours (i.e. red and orange). The lighting variance is also observed over different parts of the water's view. Nevertheless, this illumination variance over the water's region may have an effect on partitioning the central slice from the bottom slices. However, as long as we are using the RGB colour space, the illumination variance can always cause a risk of colour changes. To resolve this we need to use different colour spaces which take the illumination changes into account (i.e. L*a*b*). It is important to note that the increase in illumination does not necessarily decrease the potentiality of having a good stabilisation. However, the lighting level needs to remain the same during the whole tracking period and any changes in the lighting level can cause the tracking to perform differently. The indicated black dot on the frames' sequence shows a stable performance but it was less accurate than the performance in the previous test (see Figure 6.17). Nevertheless, as mentioned earlier, this inferior performance could be due to the increase in the surrounding illumination and decrease of colours' saturation. This was highly reflected over the second, fourth, sixth, eighth and tenth frames where they all look to have a darker illumination. Due to having a low illumination, the black dot shows a low stabilisation performance over these frames. This also indicates that the captured reference could be an illuminated frame.

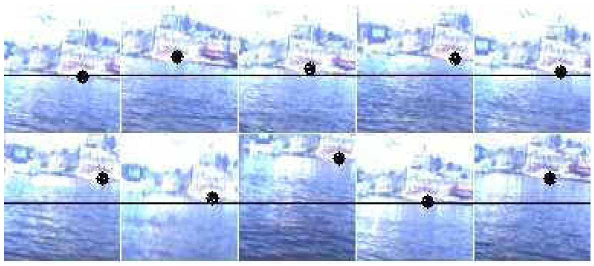


Figure 6.18 - Stabilisation process in the second test of the fifth experiment

Table 6.7 presents the pixel displacements, standard deviations and their average value over the 5 tests of the fifth experiment. In contrast to some previous examples, this experiment was free from any negative values. The gained average is 5.18 which is a close number to the second experiment's average number (4.51). Figure 6.18 shows some variance on the lighting level over the captured frames. However, the gained average value is less than the retrieved average numbers over the first, third and fourth experiments, which makes it the second best applied experiment. However, we have learnt that it is not necessarily true that when the lighting level varies, the stabilisation also becomes misled. Figure 6.18 shows some of the captured frames that are too lightened and a few others are more darken but the stabilisation over all frames performed similarly.

Table 6.7 - Pixel displacement values of the fifth sea experiment

Experiment 5	**Test 1**	**Test 2**	**Test 3**	**Test 4**	**Test 5**
Pixel Displacement	28	17	11	34	47
	24	20	15	14	33
	19	13	9	34	37
	30	19	16	24	30
	31	23	25	35	32
	24	15	23	28	30
	24	17	14	34	35
	26	18	24	29	31
	24	21	22	37	34
	24	15	14	22	47
Standard Deviation	3.50	3.04	5.73	7.26	6.39
Average	5.18				

6.3 Kite Experiments

After excluding the SSP over the sea experiments, we now bring the SSP method back to test the entire system in a real world environment. As explained earlier, the Kite's experiments are applied in Indoor and Outdoor environments. Each of these environments can present different environmental issues to demonstrate how well the system is able to resist those environmental influences over the kite's performance. However, as in the laboratory experiments, here we use the SSP technique to determine its ability to perform in a real world environment. In contrast to the laboratory experiments, we are aware of the possible colour variations which can occur during the tracking period. Moreover, in the laboratory environment, the precision of swinging over a specific slice is much more than in a real world scenario, for example, our SSP may choose a specific slice but due to the windy conditions, we may end up stabilising over a different slice. To make the comparison more reliable, we still need to use the same PC specifications for the kite's experiment. As is well known, if any of the technical specifications change, then the calibration process needs to be reconfigured. This reconfiguration reflects on the patch size selection and the SSP performance. Figure 6.19 shows the two monitored areas for both Indoor and Outdoor environments, however, the Indoor environment looks more illuminated. This high illumination is not always constant, as it is related to the main door status. For example, if the main door is open, we obtain more sun light which reflects on having a more illuminated environment and the opposite door's status causes less illumination. By having the main garage's door in constant use, the risk of a lighting variation during the tracking period is always possible as long as the RGB colour space is used. However, the effect of the sun light variance becomes less towards the evening and electrical lights will be used to light the surrounding area. The main purpose is to assess the system in the real world environment with variations in light and wind conditions. Therefore, the day time period would be our most preferred time to examine the system against the inconsistency. The experiments in the outdoor environment aimed to resist both lighting and wind variations. As Figure 6.19 shows, the Outdoor area is surrounded by two walls which can increase the region similarities over the top slices. The aim is to let the system choose the best slice and run the stabilisation after locating various colourful objects with various positioning over the central region. However, the surrounding features such as the drain and the black spots can have a significant influence on the appearance comparison technique. Unlike sea's experiments, the time consumption and the difficulties of the working environment led to decide on running single experiment over each view.

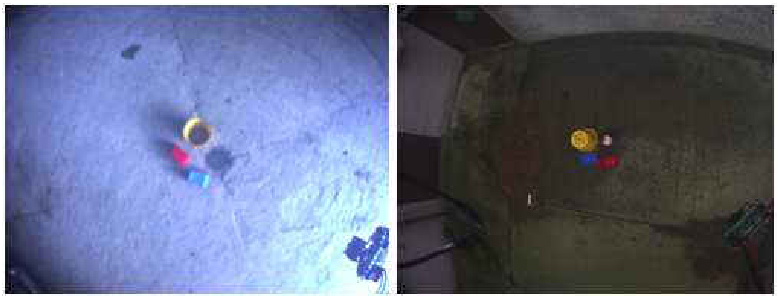


Figure 6.19 - Indoor (left) and Outdoor (right) environments are monitored from the platform's camera

6.3.1 Image Distance Coverage

To start the experiments, we would need to have a visual overview of the region's size that the platform's swing will cover. However, to examine this covering, we allow the platform to swing without enabling the stabilisation process. Figure 6.20 demonstrates the region's size covered during the platform's swing in the Indoor experiment. The size of cover can visually be observed by comparing the captured snap shot (left hand side image) with the frames' sequence. For instance, looking at the third frame (sequence counted from top left to bottom right), the drain is located in the top left hand position, but it completely disappeared over the seventh frame. The first snap shot (Figure 6.20 – left hand side image) shows that the yellow object is located on the left, the drain on the right and therefore, when the yellow object appears mostly on the right hand side of the seventh frame, it means the drain was far away from the right hand side image's boundary. This shows how large the region's size was that was covered by the platform's swing. However, the region's size coverage depends on both the pendulum's length and our first push. Consequently, we can increase the pendulum's size if the region's size needs to be increased. Figure 6.20 shows 11 captured frames, where one is captured while the platform was still and the other ten frames were captured while the platform started to swing without making any servos movement. The purpose of capturing these frames is to visually observe the camera's coverage that we have by the platform's swing. Nevertheless, the covered region's size shown by the frames' sequence of Figure 6.20 looks good enough to start the experiments with. Figure 6.20 shows three colourful substances and a drain in the central part, which are highly distinct from the background.

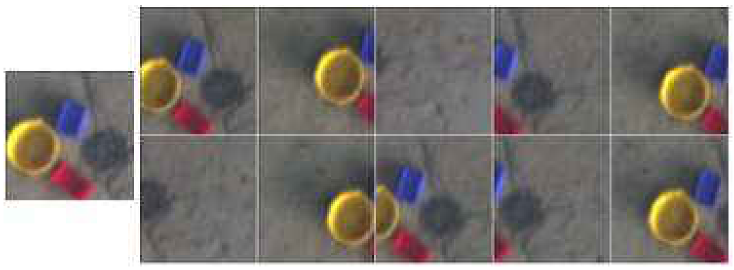


Figure 6.20 - The left hand side shows the first captured image before the platform's swing and the right hand side frames' sequence shows the region size covered in the Indoor environment

The process of defining the region's size coverage was also repeated over the Outdoor environment. Figure 6.21 demonstrates this region's size cover by the captured frames' sequence. However, the substances are appearing in some frames and completely disappearing in others. Also, the position of some substances changes dramatically, for instance, comparing the sixth and the seventh together, the position of the yellow substance changes from the top right in the sixth frame to the bottom left of the seventh frame. This indicates the large sized region the platform's swing has covered.

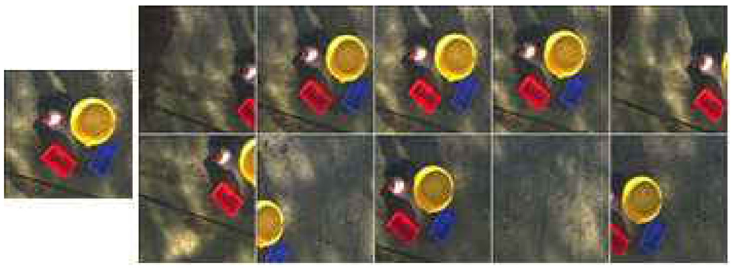


Figure 6.21 - The left hand side shows the first captured image before the platform's swing and the right hand side frames' sequence shows the region size covered in the Indoor environment

We will be applying three different experiments for both the Indoor and Outdoor environments. As earlier explained, the difference between the experiments is in the way objects are selected and positioned.

6.3.2 Indoor Experiments

6.3.2.1 First Experiment

Figure 6.22 shows the view of our first experiment in an Indoor environment. The used objects are the yellow, blue and red plastic substances plus the drain, which makes them four clustered substances in the central region.

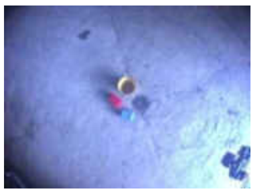


Figure 6.22 - Area view for the first Indoor experiment

We start by running the SSP over the current view. Table 6.8 shows the gained SSP results where it presents the sixth slice with the lowest and the second is the highest retrieved SSP.

Table 6.8 – Indoor first experiment DC values

Slice	1	2	3	4	5	6	7	8	9
DC	401	461	312	120	106	94	256	331	344

We now use both the second and sixth slices to observe the difference in performance between the two scenarios. The visual overview shows the fourth slice having all the colourful substances clustered in the central region and therefore, we will run the stabilisation over the fourth slice to compare its stabilisation potentialities against the sixth slice.

Figure 6.23 shows the frames' sequence of the stabilisation performance over the second slice (worst slice). The yellow substance (yellow bin) is the only major region variance within the shown sequence; therefore, we will use this substance (yellow bin) position to determine the stabilisation performance. The object was displaced by almost half of the frame's distance. We may not notice the importance of these changes but if the targeted object(s) were smaller in size or more substances existed these changes will become more observable. However, we might change the SSP's ultimate outcome if more substances were added to this frames' sequence.

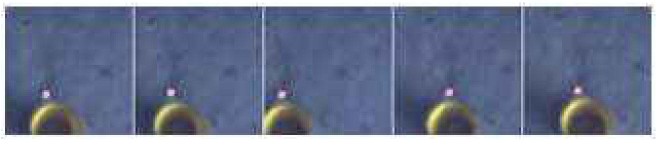


Figure 6.23 - Stabilisation over the second slice of the first Indoor experiment

To evaluate the stabilisation in a more precise procedure, we will use a common area (or point) which can be seen in all (or most) of the frames' sequences. The pink dot is an indication of a common point over all captured frames (see Figure 6.23). Table 6.9 shows the pixel positioning of the pink point over the sequence (we will use the small sized images for this evaluation process). The standard deviation of all positions is also calculated to determine the variation between the gained positions.

Table 6.9 - Indoor first experiment pixel positions over the worst slice

Position	26	23	13	32	33
Standard Deviation	8.08				

The fourth slice was chosen according to its visual image's attributes. As earlier explained, the substances on the fourth slice are clustered in the central region, which made it the most divergent slice from the region/colour variance point of view. However, the fourth slice has also gained a low DC value. Figure 6.24 shows the stabilisation performance over the fourth slice. We can easily observe the visual differences and the stabilisation performance between this frames' sequence (Figure 6.24) and the captured sequence over the second slice (see Figure 6.23) by observing the positions of the yellow bin in both sequences. In contrast to a previous scenario (see Figure 6.23), in Figure 6.24 (excluding the last frame), the yellow bin looks to have less variation than in the previous state. The last frame shows a slight change in the positioning with the objects shifted to the left hand side location. However, this does not always mean that the stabilisation behaved incorrectly but it could be due to the camera's overshoot during the snap shot. Moreover, having one out of five frames not being stabilised is an acceptable case but having the variations on the objects' positions over the entire sequence (i.e. Figure 6.23) is definitely a sign of a non-stabilised condition.

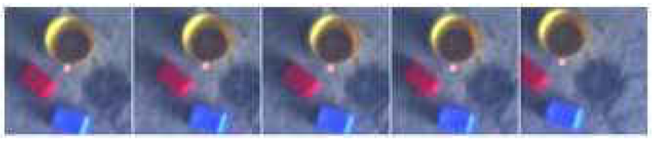


Figure 6.24 - Stabilisation over the fourth slice of the first Indoor experiment

As stated previously, to obtain a more precise evaluation, we have indicated a pink dot below the yellow bin. Table 6.10 shows the pink dot's gained positions over the fourth slice. The standard deviation was also calculated to determine the variation of the object positioning throughout the whole sequence. However, the gained standard deviation is defined as 6.18 and that is less than the gained number over the second slice test. This indicates how the region/colour variations affect the ultimate outcome.

Table 6.10 - Indoor first experiment pixel positions over the fourth slice

Positions	35	40	38	35	24
Standard Deviation	6.18				

We now move to test the stabilisation over the best gained slice (sixth slice). Figure 6.25 shows the frames' sequence of the stabilisation process over the best found slice (sixth slice). The blue substance is our main object which influenced the variation of this slice. However, part of the red substance is also included within the frames but our patch is designed to capture and iterate through the central X axis (see Chapter 4) and the possibility of capturing that red area is extremely unlikely. Nevertheless, even the blue substance will not be fully within the reference patch but from the SSP outcomes, we can determine that this small influence of the blue substance is enough to promote it as a best image's slice. Consequently, we have located the pink dot below the blue substance to determine its precise positioning through the frames' sequence. This shows us how well the stabilisation has performed over the best slice (sixth slice).

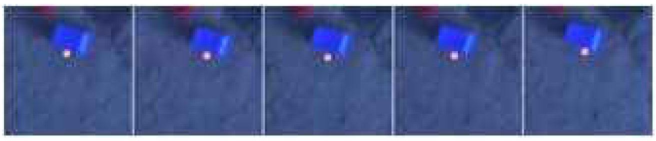


Figure 6.25 - Stabilisation over the sixth slice of the first Indoor experiment

The pink dot's positions and their standard deviations are presented in Table 6.11. The gained standard deviation is 2.50, which is less than the retrieved outcome over the second and fourth slices. This indicates how well the stabilisation was achieved in comparison to the previous two scenarios (second and fourth Slices).

Table 6.11 – Indoor first experiment pixel positions over the best slice

Positions	34	39	34	33	33
Standard Deviation	2.50				

6.3.2.2 Second Experiment

Our second Indoor experiment differs only in the way the same colourful substances are placed. After we had them all clustered in the central region (first Indoor experiment), we now have them aligned horizontally. Figure 6.26 shows how the colourful substances are aligned with the black drain over the central region. The purpose is to observe if the aligned objects will encourage the SSP to select their slice as the best one to stabilise over.

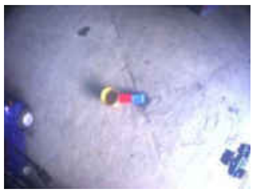


Figure 6.26 - Area view for the second Indoor experiment

Table 6.12 shows the retrieved SSP outcome over the view shown in Figure 6.26. The fifth slice gained the lowest and the first gained the highest DCs. The experiment will run the stabilisation over both slices.

Table 6.12 – Indoor second experiment DC values

Slice	1	2	3	4	5	6	7	8	9
DC	280	272	229	209	101	157	190	221	182

Figure 6.27 shows the frames' sequence captured during the stabilisation process over the first slice (worst slice). To indicate some of the common areas, we have drawn a blue circle around one region (second and fifth frames) and a pink dot around another visible region (first, third and fourth frames). The reason for indicating two different regions is because of the inability to find a single region being visible throughout the sequence. However, having the regions appearing in some frames and disappearing in others is an indication of a poor stabilisation performance over the first slice. This poor performance could be due to the low region/colour variations throughout

the slice. This low region/colour variance is common in some other image's slices (i.e. seventh, eighth, ninth), but the first slice is less illuminated than the others which might be the reason as to why SSP selected it as the worst slice. Nevertheless, we need to choose one indicator to define its positions throughout the sequence. As with previous examples, these positions will be used for the evaluation purpose. However, the second indicator (i.e. blue circle) will be used to help with defining how far the main indicator (i.e. pink dot) was shifted. The distance between the two indicators (blue circle and pink dot) needs to be determined from Figure 6.26.

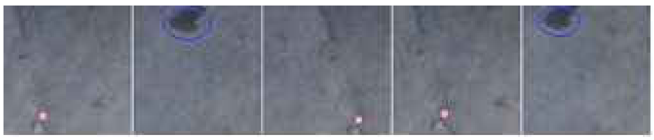


Figure 6.27 - Stabilisation over the first slice of the second Indoor experiment

Table 6.13 shows the pink dots' positions over the frames' sequence (if applicable). However, the negative values are an indication of having the pink dots out of the frame's boundaries, which resulted in obtaining a high standard deviation value over the applied experiment.

Table 6.13 – Second experiment pixel positions over the worst slice

Positions	23	-40	51	28	-49
Standard Deviation	44.38				

Figure 6.28 shows the frames' sequence of the stabilisation procedure over the fifth slice (best slice). As we expected, the aligned objects are all appearing within the image space and their positions are almost identical in the entire sequence. There are a few shifting from right to left and left to right but ultimately the images were well stabilised. The objects' colours and some of the ground's fractures made this slice the most variable in terms of its colours' contents. However, the difference in performance over the worst and best slices (first and fifth) is highly obvious, which also indicates the high viability of the SSP as a prerequisite condition for the

stabilisation procedure. Figure 6.28 shows the indicated pink point as a selected common point over the whole sequence. The pink dot is located below the yellow bin and it is used for the purpose of evaluation.

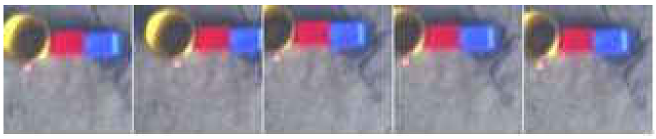


Figure 6.28 - Stabilisation over the fifth slice of the second Indoor experiment

Table 6.14 shows the retrieved pixel positioning of the pink dot over the whole sequence. However, the retrieved standard deviation is 6.37, which is a huge reduction when it is compared against the stabilisation process over the first slice (see Table 6.13).

Table 6.14 – Second experiment pixel positions over the best slice

Positions	14	24	9	9	10
Standard Deviation	6.37				

6.3.2.3 Third Experiment

Figure 6.29 shows the third and last indoor experiment. The used substances are different to the ones used in the last two experiments. We used a few wood pieces with colours that are quite close to the background colouring. The aim is to test the system's abilities against these sorts of views.

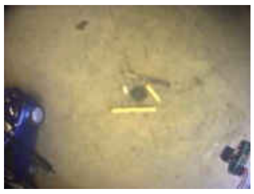


Figure 6.29 - Area view for the third Indoor experiment

Table 6.15 shows the retrieved SSP outcomes over the experimental view (see Figure 6.29). The sixth slice gained the lowest and the first gained the highest DCs value. However, looking at the retrieved numbers, the lowest found SSP is 409 DCs which is a high SSP outcome. Nevertheless, if we look back at all previous experiments (laboratory and real world), the best stabilisation performances were mostly over those slices with less than 140 or 150 DCs. However, as with the Door image experiment (see Chapter 5), if both the best and worst slices have a high DCs value, then the expectation is to have a poor stabilisation performance over both slices.

Table 6.15 – Indoor third experiment DC values

Slice	1	2	3	4	5	6	7	8	9
DC	785	684	653	410	457	409	712	703	771

We now process the stabilisation technique over both slices (first and sixth). Figure 6.30 shows the images' sequence retrieved during the stabilisation technique over the first slice (worst slice). Pink dots indicate the found common region across the sequence. However, the pink dot is shown on 4 out of 5 frames, which is an indication of missing that common region in one of the captured frames. Due to the difficulties of finding further common regions, to evaluate our data, we will use the position "0" for the frame with the missed region.

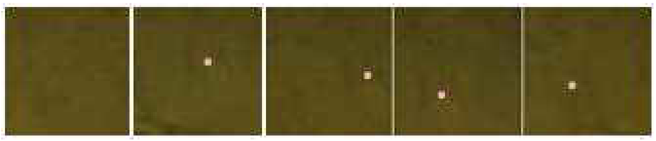


Figure 6.30 - Stabilisation over the first slice of the third Indoor experiment

Table 6.16 shows the pixel positions of the pink dots over the whole sequence. The gained standard deviation is 20.16, which we will compare against the experiment over the best slice.

Table 6.16 – Third experiment pixel positions over the worst slice

Pixel Displacement	0	41	54	26	26
Standard Deviation	20.16				

Figure 6.31 shows the frames' sequence of the stabilisation process over the sixth slice (best slice). Pink dots are located below a small black region on the wood stick to examine the performance of our stabilisation technique over the best slice (sixth slice). However, the indicated common region appeared in 4 out of 5 frames. As in the previous experiment, we will use the position "0" for the frame with the missed region.

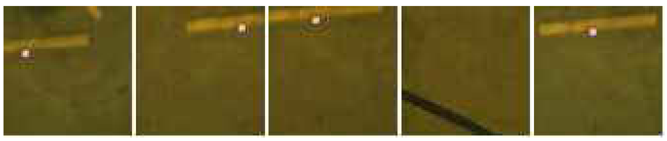


Figure 6.31 - Stabilisation over the sixth slice of the third Indoor experiment

Table 6.17 shows the pink dots' positions over the whole sequence. The retrieved standard deviation is "21.17", slightly higher than the retrieved standard deviation over the first slice (see Table 6.16). This approves the failure of the SSP over this image (see Figure 6.29).

Table 6.17 - Second experiment pixel positions over the best slice

Positions	13	56	26	0	33
Standard Deviation	21.17				

However, as explained earlier, if the best slice still has a high DCs value, it signifies that the whole image (or view) does not have potential for the stabilisation purpose. Nevertheless, the comparison between the best and worst slice will still be processed to observe the possible difference between the two scenarios.

6.3.2.4 FFT Results

As for laboratory experiments, we would need to run the FFT analysis for every retrieved pixel displacement graphs in Indoor experiments. The results will be compared against the DCs values. Table 6.18 shows both the DCs and their FFT analysis for every image slice of the Indoor experiment.

Table 6.18 - DC and FFT analysis numbers for every image's slices of the Indoor environment

EXP	Slices																	
	1		2		3		4		5		6		7		8		9	
	DC	FFT	DC	FFT	DC	FFT	DC	FFT	DC	FFT	DC	FFT	DC	FFT	DC	FFT	DC	FFT
1	401	52.0	461	110.0	312	31.9	120	19.2	106	19.4	94	18.2	256	40.7	331	36.0	344	39.5
2	280	62.1	272	47.2	229	43.6	209	38.3	101	26.8	157	38.1	190	38.2	221	37.3	182	38.5
3	785	266.3	684	257.8	653	190.8	410	151.7	457	142.5	409	95.2	712	267.2	703	226.7	771	304.4

Looking at Table 6.18, the lowest FFT analysis values were found over the slices with the lowest DC value. This shows that the outcome for the FFT analysis was quite close to our DC outcomes. This indicates a great match between the DC and FFT analysis. However, as the FFT is a well-known and widely used technique, we can use this matching as a way to back up our DC analysis from the research point of view.

6.3.2.5 Angle's View

As explained earlier, the angle's view can significantly influence the tracking performance. The reason for having a lens with a larger angled view is to cover the most possible ground space to avoid the possibilities of none overlapping between the consecutive frames. To test how the angle's view affects real world experiments, we apply an experiment using a lens with a narrower angle view (6mm) to appreciate the effect of using a wider angle view lens (3.5mm). However, as we are testing the effect of the angle's view only, we will exclude the use of SSP and place our colourful substances over the central slice.

Figure 6.32 shows the captured views using both lenses, but it is visually obvious that the 3.5mm lens has captured a larger image's space. This makes the possibility of the frames overlapping higher.

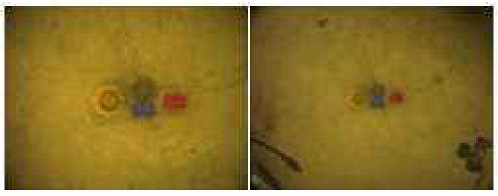


Figure 6.32 - Views from 6mm lens (left) and 3.5mm lens (right)

Figure 6.33 shows the snap shot of the view and the frames' sequence of the stabilisation process using the 6mm lens. However, to observe the difference more carefully, we have captured 10 frames over the whole sequence. We have drawn a blue circle over the interior part of the yellow bin to track it over the sequence. Nevertheless, the blue circle (common region) appeared on 9 out of 10 frames. The blue circle location on the first snap shot (Figure 6.33 - left hand side image) is the reference position. The blue circle positions in the frames' sequence shows the stability of the camera during the platform's swing.

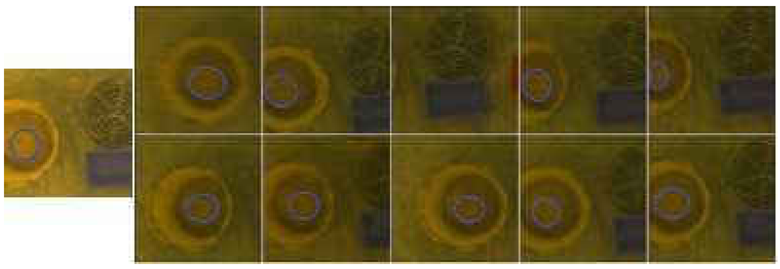


Figure 6.33 - Stabilisation with 6mm lens over the central slice

Table 6.19 shows the pixel positioning of the indicated blue circle. The standard deviation is also calculated to compare it against the subsequent experiment (3.5mm lens experiment). The gained standard deviation is 14.17. However, having clustered colourful objects in the central region decreases the possibility of the stabilisation failure even if a narrow lens was used.

Table 6.19 - Retrieving pixel positions using 6mm lens

Positions	37	10	0	8	5	35	21	39	14	12
Standard Deviation	14.17									

Figure 6.34 shows the repeated process of the stabilisation technique using the 3.5mm lens. As with previous experiments (Figure 6.33), the blue circle was also drawn in the interior region of the yellow bin. This indicates how well the wider angle view affected the stabilisation technique. However, the blue circle (common region) is also missed out in some captured frames (i.e. fourth frame), but the overall performance demonstrates a better stabilisation outcome.

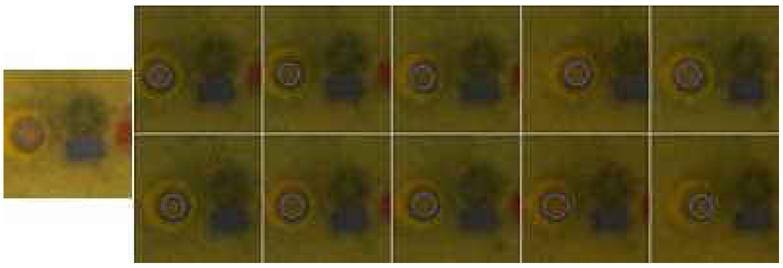


Figure 6.34 - Stabilisation with 3.5mm lens over the central slice

Table 6.20 shows the pixel positioning of the blue circle (common region) across the sequence. The gained standard deviation is 5.57 and that presents more of an enhancement of the stabilisation performance than the previous experiment (see Table 6.19).

Table 6.20 – Retrieving pixel positions using 3.5mm lens

Positions	14	14	16	30	21	21	14	17	19	27
Standard Deviation	5.57									

Nevertheless, this improvement can be due to the larger area this lens (3.5mm lens) has covered, which can be notified by observing the difference of both blue circles (Figures 6.33 & 6.34). A wider angle view reflects on having more possible feature variations, resulting in obtaining a better SSP outcome.

6.3.3 Outdoor Experiments

As in Indoor environments, we will now repeat similar experiments in a complete Outdoor environment. Referring back to Figure 6.1 (Middle Image), it shows the platform is hanging between the wall and a tree, challenging both the light and wind condition variance.

6.3.3.1 First Experiment

Figure 6.35 shows the view of our first Outdoor experiment. As in the first Indoor experiment, the colourful substances are all clustered in the central region. However, as we had 3 colourful features and a drain which represented our fourth feature in the Indoor environment, a plastic rubber cone sign is added to the three substances in our Outdoor experiments to equalise the number of clustered features for both environments. In our Outdoor space, the top and left regions are concrete walls which increase the region variations dramatically. This would add extra difficulties to our ground colour invariance. However, due to earlier rain, the ground colouring does vary in some areas, while the wet regions appear darker than the dried ones. As explained earlier, the lighting variances are due to having variations in both the sun light and the electrical light, which goes on and off every so often. This light variance highly affects the colour variation of the similar object(s).

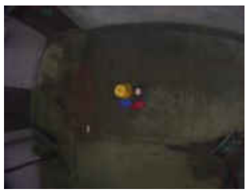


Figure 6.35 - Area view for the first Outdoor experiment

Table 6.21 shows the retrieved SSP numbers over the demonstrated area (see Figure 6.35). We will run the stabilisation technique over the best and worst slices (fifth and first slices) to observe how well this technique matches our expectations of the SSP in a complete Outdoor environment.

Table 6.21 - First Outdoor experiment DC values

Slice	1	2	3	4	5	6	7	8	9
DC	389	294	284	152	122	184	158	146	145

Figure 6.36 shows the frames' sequence of the stabilisation process over the first slice (worst slice). The blue circle shows the common region over the whole sequence. However, taking the high colour repetition into account, to some extent the blue circle shows good stabilisation achievement.

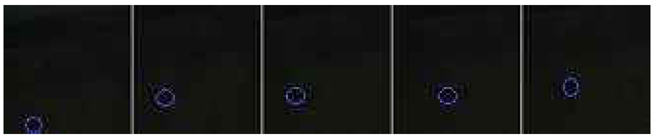


Figure 6.36 - Stabilisation over the first slice of the first Outdoor experiment

Table 6.22 shows the pixel positions and their standard deviation of the common region (blue circle) over the whole frames' sequence (see Figure 6.36).

Table 6.22 – Outdoor first experiment pixel positions over the worst slice

Positions	18	18	18	30	26
Standard Deviation	5.65				

Figure 6.37 shows the frames' sequence captured over the fifth slice experiment. Unlike the previous scenario (Figure 6.36), the substances' positions appear to be stable over both the X and Y axes.

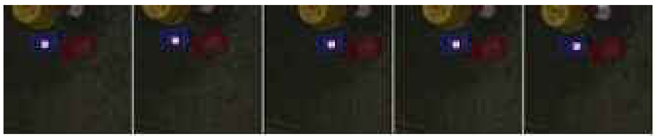


Figure 6.37 - Stabilisation over the fifth slice of the first Outdoor experiment

Table 6.23 shows the pixel positions of the common region (blue circle) over the fifth slice. However, the calculated standard deviation is now 5.58, which is lower than the first slice (5.65) outcome.

Table 6.23 - Outdoor first experiment pixel positions over the best slice

Positions	24	24	36	34	28
Standard Deviation	5.58				

Due to having a small difference between both standard deviations, a similarity in the performance of both scenarios is indicated. Nevertheless, this similarity is mostly due to verifying the blue circle positions over the X axis only. However, if we also take the Y axis into account, the positions' instability for the first slice experiment (Figure 6.36) becomes very apparent.

6.3.3.2 Second Experiment

As in the Indoor experiments, for our second experiment, we align the used colourful substances over the horizontal space. As usual, the SSP method will determine which slice is the best to use. Figure 6.37 shows the experimental area and the way substances are aligned. However, the area looks more illuminated than how it previously was (see Figure 6.38) which could be due to either the sun's position or the on and off of the electrical light which we mentioned earlier.

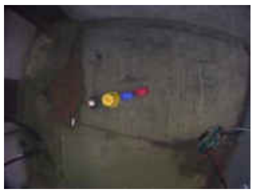


Figure 6.38 - Area view for the second Outdoor experiment

Unlike the previous experiment where the first slice was chosen as the worst, in this experiment the ninth slice is the worst selected slice (see Table 6.24). However, the change of illumination may have changed the ultimate SSP outcome. Nevertheless, in a previous experiment, the first slice was appointed the second highest DC, which still counted as one of the worse slices to stabilise over. We are expecting the best slice to be the one with the aligned substances included. However, according to the numbers shown in Table 6.24, the fifth is the best chosen slice from the SSP point of view.

Table 6.24 - Second Outdoor experiment DC values

Slice	1	2	3	4	5	6	7	8	9
DC	511	392	331	255	105	299	309	399	647

Figure 6.39 shows the frames' sequence of the stabilisation process over the fifth slice (best slice). To determine the precise evaluation, we have located the pink dot over the yellow bin's base.

However, the frames' sequence shows a gradual shifting of the substances as we near towards the end. This shift can be observed clearly by monitoring the positions of the plastic rubber cone sign which disappears by the end of the sequence.

Due to mainly having the colourful substances on the top part of the sequence, the ground colour repetition over the central and the lower regions could be the most influencing point over the shifting progression. Due to setting a time limit for the stabilisation period, we are unaware of how far this shift can proceed and which substances will also disappear from the image's space. However, a camera's overshoot may compensate this shift and return the camera back to the origin.

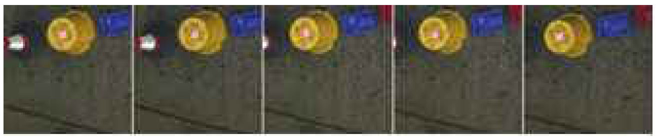


Figure 6.39 - Stabilisation over the fifth slice of the second Outdoor experiment

Table 6.25 shows the pixel positions of the common region (pink dots) across the frames' sequence over the fifth slice.

Table 6.25 - Second Outdoor experiment pixel positions over the best slice

Positions	32	33	24	23	21
Standard Deviation	5.50				

Figure 6.40 shows the frames' sequence of the stabilisation process over the ninth slice (worst slice). However, it looks identical to the first slice of the previous experiment (see Figure 6.36) but the absolute darkness is dominating and makes the process of finding the common region impossible. Consequently, the evaluation of the stabilisation process over this slice is unattainable but we noticed the camera's physical maneuver went too far to be called "stabilised".

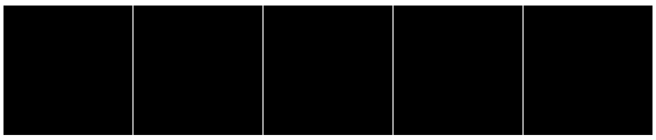


Figure 6.40 - Stabilisation over the ninth slice of the second Outdoor experiment

6.3.3.3 Third Experiment

Figure 6.41 shows the third Outdoor experiment's view. As with the third Indoor experiment, we used a few wooden pieces to replace the plastic colourful substances. The new replaced objects would decrease the saturation of the clustered substances in the central region. This reduction would trim down the region's uniqueness, causing some side effects to the ultimate SSP outcome.

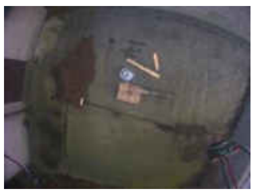


Figure 6.41 - Area view for the third Outdoor experiment

Table 6.26 shows the retrieved SSP numbers over the demonstrated area (see Figure 6.41). The stabilisation technique will be used to observe how well it will perform over the best and worst slices (fifth and sixth slices). However, the majority of slices gained quite similar DCs, which indicates a large reduction over the features' uniqueness across the image's space.

Table 6.26 - Third Outdoor experiment DC values

Slice	1	2	3	4	5	6	7	8	9
DC	268	273	274	207	202	425	222	239	224

Figure 6.42 shows the frames' sequence of the stabilisation over the best slice (fifth slice). The blue circles are drawn over the common region across the sequence. Figure 6.42 shows the existence of the common region during the whole frames' sequence. However, the common region shows the camera's shift from central to the most left region. Depending on how the objects are aligned, a shift over a single axis may also reflect some shifting proportion over the opposite axis. Yet, this depends on how the objects are aligned to the way the platform is swinging.

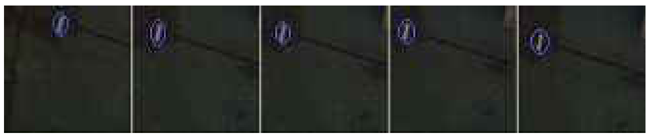


Figure 6.42 - Stabilisation over the fifth slice of the third Outdoor experiment

Table 6.27 shows the pixel positions of the common region (blue circle) over the fifth slice (see Figure 6.42). The calculated standard deviation is 8.57, although the first pixel position ("32") looks to be a camera's overshoot. This may have boosted the calculated standard deviation further up.

Table 6.27 - Third Outdoor experiment pixel positions over the best slice

Positions	32	16	13	11	13
Standard Deviation	8.57				

Consequently, if we exclude the first value ("32") from Table 6.27 and recalculate the remaining 4 values, the new standard deviation becomes 2.06 (see Table 6.28). This shows a huge reduction from the previously calculated standard deviations (see Table 6.27). This reduction illustrates the enormous effect of the camera's overshoot on the ultimate outcome.

Table 6.28 - Excluding one standard deviation value from the third experiment and calculate the new average

Positions	16	13	11	13
Standard Deviation	2.06			

Figure 6.43 shows the frames' sequence captured over the sixth slice (worst slice). As with the previous frames' sequence (Figure 6.43), we have drawn blue circles around the common region throughout the sequence. However, the frames' sequence shows a high level of instability over the vertical axis, but our system is designed to recover only horizontally. Nevertheless, this vertical instability could be due to the horizontal instability by having dissimilar objects aligned against the way the platform swings.

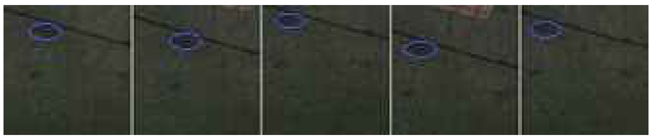


Figure 6.43 - Stabilisation over the sixth slice of the third Outdoor experiment

Table 6.29 demonstrates the pixel positions and their standard deviation value over the sixth slice scenario. However, this value looks smaller than the gained standard deviation in Table 6.27 (best slice), but it becomes larger if we exclude the overshoot value over the first captured frame (see Table 6.28). Nevertheless, this clarifies a better stabilisation performance over the best slice.

Table 6.29 - Third Outdoor experiment pixel positions over the worst slice

Positions	23	28	16	17	16
Standard Deviation	5.33				

6.3.3.4 FFT Results

As in the Indoor experiments, we will run the FFT analysis for every retrieved pixel displacement graph in the Outdoor experiments. The results will be compared against the DC values of the Outdoor experiments. Table 6.30 shows both the DC and their FFT analysis for every image slice of the Outdoor experiment.

Table 6.30 - DC and FFT analysis numbers for every image's slices of the Outdoor environment

EXP	**Slices**																	
	1		**2**		**3**		**4**		**5**		**6**		**7**		**8**		**9**	
	DC	**FFT**	**DC**	**FFT**	**DC**	**FFT**	**DC**	**FFT**	**DC**	**FFT**	**DC**	**FFT**	**DC**	**FFT**	**DC**	**FFT**	**DC**	**FFT**
1	389	130.0	294	132.0	284	114.9	152	56.9	122	38.1	184	57.2	158	42.7	146	61.3	145	34.0
2	511	208.3	392	167.8	331	164.5	255	156.8	105	31.4	299	139.8	309	164.0	399	169.0	647	253.8
3	268	69.3	273	72.4	274	47.3	207	46.4	202	71.4	425	86.0	222	35.3	239	50.9	224	24.6

Table 6.30 shows a complete match on the second, a partial match on the third and no match on the first experiments in allocating the smallest and largest FFTs over the same slices with the smallest and largest DCs. In the third experiment, the largest FFT value refers to the same slice with the largest DCs, but the smallest FFT refers to the ninth slice, which also has one of the lowest DC numbers. However, the first experiment is the only one whereby both the smallest and largest FFT analysis were not allocated over slices with the smallest and largest DCs. Nevertheless, the smallest and largest FFTs were also allocated over slices which held small and large DCs values. Consequently, we are also able to depend on FFT analysis for the stabilisation technique.

6.3.3.5 *Angle's View*

As with the Indoor environment, we now repeat a similar experiment of using a narrower lens (6mm) to observe the effect of the wider angle's view over the stabilisation performance in the Outdoor environment. Similar to previous scenarios, we exclude the use of SSP and place some colourful substances over the central slice. Figure 6.44 shows the captured views using both lenses. It shows how the 3.5mm lens captured a larger image's space which increases the overlapping possibilities. The captured image with the wider lens (3.5mm) appears more illuminated. However, a more condensed view could have the influence of lessening the image's illumination level. Our prerequisite was to ensure the software and the camera's settings are identical in both Indoor and Outdoor environments, to ensure the credibility of our comparison technique.

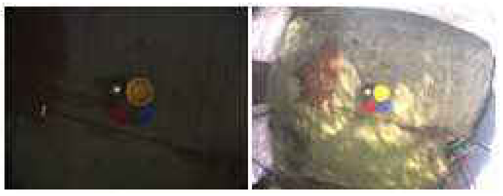


Figure 6.44 - Views from 6mm lens (left) and 3.5mm lens (right)

Figure 6.45 shows both the reference snap shot (left hand side image) and the frames' sequence, which illustrates how well the stabilisation was achieved. We can visually observe the fact that the fourth frame is the only stabilized frame over the whole sequence. The rest of the frames show a background region completely, which indicates how far the camera was mismatched and directed away from the origin. We also cannot judge if the fourth frame was stabilized or if it just happened to be captured when the camera was above the origin. Loss of origin can also be due to having another competitive local minimum within the image space but this sequence demonstrates a total instability scenario. We can observe this instability by observing the ground's fractures where they differ from some frames to others.

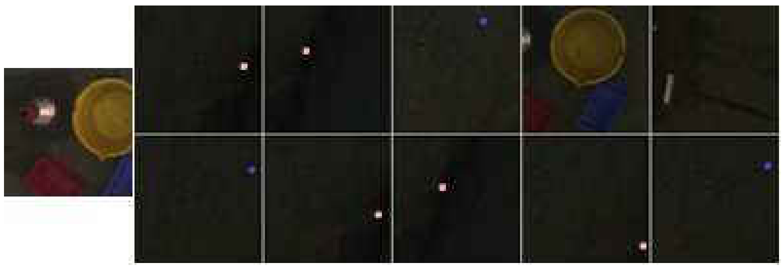


Figure 6.45 - (Stabilisation using 6mm lens). Different colour indicators showing different ground's fractures

However, there are two indicators shown over this frames' sequence (Figure 6.46 - pink and blue dots), where each refers to a common region across the sequence. Figure 6.46 shows both common regions are visible over the sequence's seventh frame. The pixel distance between the two indicators presents the large instability reached with this sequence, but we choose the pink point indicator to evaluate how well the stabilisation was performed. Nevertheless, we would need to use the other visible region (with the blue indicator) to calculate the pixel positioning of the pink dot whenever it is invisible. However, as Figure 6.46 shows, the pixel distance between the two dots is 47 pixels (on this frame's size) and therefore, this distance will be added to the blue dot position whenever the pink dot is invisible.

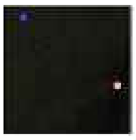


Figure 6.46 - Seventh frame showing both common regions

Due to having none of the two indicators existence over the fourth and fifth frames, we would need to calculate the possible camera's shift over the whole image's space. Figure 6.47 demonstrates the calculated distance from the blue point to the centre of two captured frames (fourth and fifth). However, we can visually clarify that the possible distance between the blue point and fourth frame looks to be approximately one complete frame's width (see Figure 6.47 (Red line)) and the distance between the blue point to the fifth frame appears to be 5 × (frame's width). On the other hand, if we attempt to define the pink point position on the fourth frame, we have to add one complete frame's width plus the blue to pink distance and for the pink point position over the fifth slice, we have to multiply 5 × (frame's width) plus the blue to pink distance.

Figure 6.47 - Approximate distance between the regions

Table 6.31 shows the pink dot pixel positions over the entire sequence. It also presents the calculated standard deviation to determine how well the camera's stabilisation was achieved.

Table 6.31 - 6mm lens experiment pixel positions outcome

Positions	56	21	94	112	372	108	58	24	62	110
Standard Deviation	100.80									

The next frames' sequence in Figure 6.48 shows the stabilisation process using a wider lens's view. Blue circles were drawn around a common region across the sequence to evaluate the stabilisation performance.

This sequence shows quite good stabilisation from the second to the ninth frames. Although, the instability is shown more on the first and last frames, this could be due to some cameras overshoot during the capturing time. This overshoot could be due to having competitive local minima within the image's space.

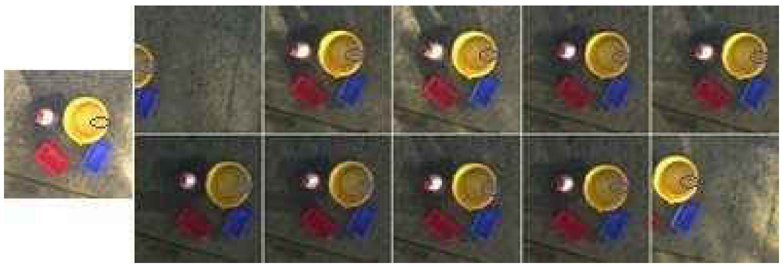


Figure 6.48 - (Stabilisation using 3.5mm lens). Blue circle indicator presents the common region across the sequence

Table 6.32 shows the pixel positions of the common region and their standard deviation over the frames' sequence (see Figure 6.48). However, using the wider angle's lens increased the stability and decreased the standard deviation from 100.80 (Table 6.31) down to 16.77 (Table 6.32). This demonstrates the effect of a large angle's lens over the ultimate outcome.

Table 6.32 – 3.5mm lens experiment pixel positions outcome

Positions	7	48	51	52	57	57	54	52	52	22
Standard Deviation	16.77									

6.4 Conclusion

Real world experiments were applied to test the performance of our designed system over different environments. The aim was to test the workability of the SSP and the hierarchical search algorithms in real world scenarios and also to sustain the laboratory's gained outcome further. Nevertheless, as earlier explained, we tested the designed hierarchical search algorithms over the Indoor, Outdoor and the Sea environments, but due to impracticality, the SSP was tested over the Indoor and Outdoor environments only (i.e. flying scenarios). However, despite the fact that the SSP was excluded, we have always chosen the central slice that holds the majority of the region variations in our sea experiments. This would decrease the necessity for such a slice selection technique.

On the other hand, choosing the central slice of our sea experiments may not consistently lead to the best selection. Table 6.33 presents the gained average of all the applied sea experiments' outcomes. It shows how some experiments gained a low result and how others have achieved a higher result. This is an indication that the central slice in some experiments led to a better stabilisation.

Table 6.33 - Average of all experiments' standard deviations

Sea Experiments				
Experiment 1	Experiment 2	Experiment 3	Experiment 4	Experiment 5
13.34	4.51	11.45	2.84	5.18
Average	7.46			

Due to the procedures' similarities, we are able to present and compare the Indoor and Outdoor experiments' outcomes simultaneously. Nevertheless, the third experiments in both scenarios have presented higher values which are an indication of the camera's instability during the experimental period. However, the camera's high instability in the Indoor third experiment was reflected in the camera's better stability in the Outdoor environment.

Table 6.34 - Averages of all Indoor and Outdoor experiments' standard deviations results over the best slices

Indoor Experiments		
Experiment 1	Experiment 2	Experiment 3
2.50	6.37	21.17
Average	10.01	
Outdoor Experiments		
Experiment 1	Experiment 2	Experiment 3
5.58	5.50	8.57
Average	6.55	

Eventually, the achieved experiments were applied to answer the question asked in chapter 1 that identifies the appropriate regions for image-based stabilisation using pixel-wise comparison in real world environments.

However, as in laboratory environments, the experiments with high region/colour variations have shown better camera stability. This shows how these variations have enhanced the camera's stability.

Chapter 7

Conclusion

7.1 Preface

The techniques and simulations developed in chapter 4 combined with the experiments and results described in chapter 5 and 6 support a positive answer to the research question: *Can appropriate regions for image-based stabilisation be selected empirically using pixel-wise comparison and simulated platform motion?*

In this chapter we summarise the content of each chapter and present the contribution made in this research. The chapter also details the limitations of this research and the possible future work that could expand it.

7.2 Summarising the content

This research started by categorising the content of the literature into two major divisions ("Input" and "Tools & Methods"). However, the "Tools & Methods" section was subcategorised as Probability Distribution, Edges and Segmentations, Contours, Features and Adaptive Methods & Filters. The listed tools/methods in each subcategory share a similar methodology that is used in their tracking systems. The research is then continued to define a methodology that fulfils the project's requirements. Various strategies were followed to implement solutions for locating a patch within the image for use in stabilisation. These strategies were chosen to attain the best possible error surfaces to achieve the most effective stabilisation. Fixed Locations, Edge Tracking, Feature Tracking and Image Smoothness are some of the implemented strategies which were applied and tested to determine their influence over the error surface generation process. However, each strategy has its strengths and weaknesses, for example, fixed locations are only good if those fixed patches happen to be located over useful features for fixation. Edge tracking also runs the risk of locating the patch over a repetitive feature region which may result in an error surface with many local minima. Feature tracking was also applied using different blob sizes which showed good results on artificial images but did not present the same quality on real world images. Also, to have more gradual changes over the error surface, the image smoothing procedure was applied to reduce the unwanted edges and keep the most effective ones, but this can easily fail if the image's blurriness increases due to some internal or external influences (i.e. environmental conditions, lens setup). Since the random selection technique was used as the main pixel selection mode, the concern remains about choosing the right range for the random distribution (patch size) and the patch position on the vertical axis. In contrast to the third chapter, the retrieved data on the fourth chapter is the result of hanging and swinging the platform in the laboratory environment. The patch size selection procedure was carried out by letting the platform swing and applying the appearance comparison between every two consecutive images. This process is repeated using different patch sizes and the displacement graph with fewer overshoots points to the most suitable patch size. We have determined that the 100×100 patch is the most appropriate patch size to be used on the 300×300 images in our experiments. However, the uEye camera can retrieve up to 1600×1200 images, so rather than locating the patch in the central position, we have implemented the Slice Selection Process (SSP) which analyses where best to locate the patch on the vertical axis. The SSP divides the image into a few slices, locates the patch and generates the pixel displacement graph for every slice. The graph with fewer overshoots is used to select the most suitable slice that the patch can be located over. To apply the appearance comparison process, Hierarchical Search algorithms were designed to find the best correlation between the reference and the current patches using Euclidean distance. Many minimisation methods (Gradient Descent and Newton's Method) navigate over gradually descending surfaces but the hierarchical searching method looks for the global minimal over the whole surface.

In the laboratory environment, experiments were done on various printed images. The SSP was used to choose the best and worst slices of each image. The stabilisations are then applied to the chosen slices and the performances are compared to determine how the SSP influenced the ultimate outcome. In addition, Chapter 4 presented useful visual analyses to explain the contrast between different images or different slices. Also, the edge distributions were investigated to verify their possible effect on the SSP and on stabilisation performance. However, to backup the SSP outcome further, the FFT technique was used to analyse the pixel displacement graphs [129]. This analysis showed a significant link between counting the Direction Changes (DC) and the FFT analysis of the displacement graphs. However, as the retrieved graphs represent the pixel displacement values, if any confliction between the DC and the FFT values appears in the graph analysis then it is safer to rely on the DC measurement, as it can reflect the platform's movement more accurately. The laboratory experiments were followed by experiments in real world environments to assess the system's performance under variable conditional environments. The aim was to test the SSP and hierarchical search algorithms in real world conditions. Both SSP and hierarchical search algorithms were tested on kite flying experiments but due to the SSP's impracticality in our boat scenario, the SSP method was excluded and the hierarchical search algorithms was tested to give evidence of the workability of this algorithm in non-flying scenarios. We always used the central slice that holds the majority of the region variations in our sea experiments therefore the necessity for such a slice selection technique is less than in the flying experiments. As in the laboratory work, the appearance investigation and the FFT analysis were also carried out in the real world experiments. However, the real world experiments and the applied analysis demonstrated a reasonable system performance in the real world projects.

7.3 Significance of the findings (research contribution)

The Slice Selection Process (SSP) is the major contribution of this research. The SSP has contributed to how to use the camera as the key sensor for continuously orienting the position over the original view. It was mainly designed to help systems that require automatic selection of image regions for active fixation.

The Slice Selection Process (SSP) uses the pixel displacement graphs to choose the image region that can best be used for active fixation. The selection technique is based on counting every graph peak and choosing the slice with the fewest peaks for camera fixation.

The research has also contributed by introducing the "hierarchical search" which aims to perform an exhaustive search with an enormous reduction in pixel processing. The hierarchical search is used to speed up the process of the Slice Selection Process (SSP).

The techniques and experiments described here show that, it is possible to perform real time stabilisation using automatically selected image regions using low computational overhead algorithms. In general, it doesn't seem to be necessary to perform complex image analysis in order to achieve this.

7.4 Limitations of the current study

This research has limitations which are observable if the system is intended to be used for a real project. The fixation is achieved in the rotation about the axes in the image plane but has not focused on scenarios where the camera rotates about its optical axis. Another limitation is due to the possible scale change during the tracking period. This study has not covered a possible solution that can be used to recover from scale changes. On the other hand, the possible changes to both the platform's rotation and the scale can influence best slice selection, therefore we would then need a better strategy that updates the best slice selection whenever such changes occur during the tracking period.

7.5 Recommendations for further work

Future work could give attention to the clarification of the current limitations. For example, as explained earlier, the camera rotating on its optical axis and the possible scale changes are the major limitations of this research. However, there are a few existing techniques that are used to determine this tracking path using a sequence of images (i.e. Dynamic Programming [2], [37] and [127]). Though, if the platform rotates dramatically, the comparison between the two paths may turn out to be impossible. On the other hand, the scale change is also considered by many literatures, for instance modifying a version of kernel's radius by ±10% can help to increase the tracking robustness. As a consequence of determining the camera rotation over its optical axis or the scale change, we may re-run the SSP method to enhance the system performance during the tracking period. Moreover, future work may also focus on converting the SSP to a more generalised feature selection method using just a single captured shot.

References

1. Allen, J.G., Xu, R.Y.D. and Jin, J.S. , *Object tracking using CamShift algorithm and multiple quantized feature spaces*, in *In Proc. 2003 Pan-Sydney Area Workshop on Visual Information Processing (VIP2003)*. 2006, Australian Computer Society, Inc: Sydney, Australia. p. 3-7.
2. Alon, J.A., Vassilis; Yuan, Quan; Sclaroff, Stan; Boston University, MA *Simultaneous Localization and Recognition of Dynamic Hand Gestures*, in *Motion and Video Computing, 2005. WACV/MOTIONS '05 Volume 2. IEEE Workshop*. 2005: Breckenridge, CO, USA. p. 254 - 260.
3. Alper Yilmaz, M.S., Javed, *Object Tracking: A Survey.* Image Processing and Computer Vision, 2006.
4. Alper Yilmaz Xin , X.L., Mubarak Shah, *Object Contour Tracking Using Level Sets*, in *Asian Conference on Computer Vision, ACCV 2004*. 2004: Jaju Islands, Korea.
5. Ambrish Tyagi and James W. Davis and Gerasimos Potamianos; Dept. of Comput. Sci. & Eng., O.S.U., Columbus, OH, *Steepest Descent For Efficient Covariance Tracking*, in *Motion and video Computing, 2008. WMVC 2008. IEEE Workshop*. 2008: Copper Mountain, CO. p. 1 - 6.
6. Aslam, T.D., *A Level Set Algorithm for Tracking Discontinuities in Hyperbolic Conservation Laws I: Scalar Equations* Hyperbolic Conservation Laws II: Systems of Equations, 1998: p. 413-438.
7. Avidan, S.M.E.R.L., Cambridge, MA *Ensemble Tracking*, in *Pattern Analysis and Machine Intelligence, IEEE Transactions*. 2005. p. 261 - 271.

8. B. Banitalebi, H.A., J-Agricultural Research center, *An Improved Nearest Neighbor Data Association Method for Underwater Multi-Target Tracking*.

9. Babenko, B.M.-H.Y.B., S.; Univ. of California, San Diego, CA, USA *Visual tracking with online Multiple Instance Learning*, in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference*. 2009: Miami, FL. p. 983 - 990.

10. Bagci, A.M.A., R.; Khokhar, A.; Cetin, E.; Illinois Univ., Chicago, IL, USA *Eye tracking using Markov models*, in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference*. 2004. p. 818 - 821.

11. Baris Sumengen, B.S.M.E.D., UC, Santa Barbara 93106, Santa Barbara, CA, USA, *Multi-scale Edge Detection and Image Segmentation*, in *European Signal Processing Conference (EUSIPCO)*. Sep. 2005.

12. Battiato, S.G., G.; Puglisi, G.; Scellato, S.; Univ. of Catania, Catania *SIFT Features Tracking for Video Stabilization*, in *Image Analysis and Processing, 2007. ICIAP 2007. 14th International Conference*. 2007: Modena. p. 825 - 830.

13. Bergener, T.D., P.; Inst. fur Neuroinf., Ruhr-Univ., Bochum, *A framework for dynamic man-machine interaction implemented on an autonomous mobile robot*, in *Industrial Electronics, 1997. ISIE '97., Proceedings of the IEEE International Symposium*. 1997: Guimaraes , Portugal. p. SS42 - SS47.

14. Björck, Å., *Numerical Methods for Least Squares Problems*. 1996.

15. Blake, M.I.a.A., *Contour Tracking by Stochastic Propagation of Conditional Density*, in *ECCV '96 Proceedings of the 4th European Conference on Computer Vision*. 1996.

16. Blake, M.I.a.A., *CONDENSATION - conditional density propagation for visual tracking*. International Journal of Computer Vision, 1998. **29**: p. 5-28.

17. Blake, M.I.a.A., *ICondensation: Unifying low-level and high-level tracking in a stochastic framework*, in *Proc 5th European Conf. Computer Vision*. 1998. p. 893-908.

18. Blake, M.I.a.A., *A smoothing filter for Condensation*, in *In Proc. European Conf. on Computer Vision*. 1998. p. 767-781.

19. Bolme, D.S.B., J.R.; Draper, B.A.; Yui Man Lui; Colorado State Univ., Fort Collins, CO, USA *Visual object tracking using adaptive correlation filters*, in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference*. 2010: San Francisco, CA. p. 2544 - 2550.

20. Borkar, M.C., V.; McClellan, J.H.; Georgia Inst. of Technol., Atlanta *A Monte-Carlo Approach for Tracking Mobile Personnel*, in *Aerospace Conference, 2007 IEEE*. 2007: Big Sky, MT. p. 1 - 10.

21. Bouguet, J.-Y., *Pyramidal Implementation of the Lucas Kanade Feature Tracker: Description of the algorithm*, in *Intel Corporation - Microprocessor Research Labs*. 2002.

22. Brethes, L.M., P.; Lerasle, F.; Hayet, J.; LAAS, CNRS, Toulouse, France *Face tracking and hand gesture recognition for human-robot interaction*, in *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference*. 2004. p. 1901 - 1906.

23. C. Harris, M.S., *A Combined Corner and Edge Detection*, in *In Proceedings of The Fourth Alvey Vision Conference*. 1988. p. 147-151.

24. Canny, J. and M.I.o.T. Artificial Intelligence Laboratory, Cambridge, MA 02139, *A Computational Approach to Edge Detection.* Pattern Analysis and Machine Intelligence, IEEE Transactions, 1986. **8**(6): p. 679 - 698.

25. Chockalingam, P.P., N.; Birchfield, S.; Electr. & Comput. Eng. Dept., Clemson Univ., Clemson, SC, USA *Adaptive fragments-based tracking of non-rigid objects using level sets*, in *Computer Vision, 2009 IEEE 12th International Conference*. 2009: Kyoto. p. 1530 - 1537.

26. CHRISTOPHER G. ATKESON1, A.W.M., STEFAN SCHAAL, *Locally Weighted Learning*. Artificial Intelligence Review, 1997: p. 11–73.

27. COHEN, L.D., *NOTE on active contour models and balloons*. 1991, Image Understanding. p. 211-218.

28. Collins, T., *Graph Cut Matching In Computer Vision*. 2004.

29. Comaniciu, D.R., V.; Meer, P.; Imaging & Visualization Dept., Siemens Corp. Res. Inc., Princeton, NJ, *Real-time tracking of non-rigid objects using mean shift*, in *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference*. 2000: Hilton Head Island, SC , USA. p. 142 - 149.

30. Dave Binding and Frédéric Labrosse; Department of Computer Science, U.o.W., Aberystwyth, *Visual local navigation using warped panoramic images*, in *In Proceedings of Towards Autonomous Robotic Systems, University of Surrey, Guildford, UK, 2006*, Taros. p. 19-26.

31. David A. Ross , J.L., Ruei-Sung Lin , Ming-Hsuan Yang, *Incremental Learning for Robust Visual Tracking*, in *Int J Comput Vis*. 2008. p. 125–141.

32. David Bolme and B. Draper, *ASEF Correlation Filters for Signal Processing and Object Detection and Recognition*.

33. Dellaert, F.F., D.; Burgard, W.; Thrun, S.; Dept. of Comput. Sci., Carnegie Mellon Univ., Pittsburgh, PA *Monte Carlo localization for mobile robots*, in *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference*. 1999: Detroit, MI , USA. p. 1322 - 1328.

34. Deza, E.D.M.M., *Encyclopedia of Distances*. 2009: Springer.

35. Dony, R.D.W., S.; Sch. of Eng., Guelph Univ., Ont., *Edge detection on color images using RGB vector angles*, in *Electrical and Computer Engineering, 1999 IEEE Canadian Conference*. 1999: Edmonton, Alta. , Canada. p. 687 - 692.

36. Doucet, A.D.F., N.; Gordon, N.J., *Sequential Monte Carlo Methods in Practice (Statistics for Engineering and Information Science)*. 2001: Springer.

37. Dreuw, P.D., T.; Rybach, D.; Keysers, D.; Ney, H.; Dept. of Comput. Sci., RWTH Aachen Univ., *Tracking using dynamic programming for appearance-based sign language recognition*, in *Automatic Face and Gesture Recognition, 2006. FGR 2006. 7th International Conference*. 2006: Southampton. p. 293 - 298.

38. Esther Horbert, D.M., Bastian Leibe; UMIC Research Centre RWTH Aachen University, Germany, *Geometrically Constrained Level Set Tracking for Automotive Applications*, in *Proceedings of the 32nd DAGM conference on Pattern recognition*. 2010.

39. Ashwani A, Susmit B, Sandeep S, Shamik S, K. Majumdar; *Object Tracking Using Background Subtraction and Motion Estimation in MPEG Videos.* Computer Vision – ACCV 2006: Volume 3852, 2006, pp 121-130.

40. Fatih Porikli, O.T., *Covariance Tracking using Model Update Based on Means on Riemannian Manifolds*. 2006, Computer Vision and Pattern Recognition: New York City.

41. Fr´ed´eric Labrosse; Department of Computer Science, U.o.W., Aberystwyth, *The visual compass: performance and limitations of an appearance-based method.* Journal of Field Robotics, 2006: p. 913–941.

42. Frank Dellaert , W.B., Dieter Fox , Sebastian Thrun, *Using the CONDENSATION Algorithm for Robust, Vision-based Mobile Robot Localization*, in *Proceedings 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Cat No PR00149*. 1999, IEEE Comput. Soc. p. 588-594.

43. Frederic Labrosse; Department of Computer Science, U.o.W., Aberystwyth, Ceredigion, SY23 3DB, *Visual Compass*, in *Taros*. 2006, Proceeding of Towards Autonomous Robotic Systems: University of Essex, Colchester, United Kingdom. p. 85-92.

44. Frintrop, S.K., Markus; Institute of Computer Science III, Rheinische Friedrich-Wilhems-UniversitÃ¤t, 53117 Bonn, Germany, *Most salient region tracking*, in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference*. 2009: Kobe. p. 1869 - 1874.

45. Fukunaga, K.H., L., *The estimation of the gradient of a density function, with applications in pattern recognition*, in *Information Theory, IEEE Transactions*. 1975. p. 32 - 40.

46. Georg Klein; Active Vision Lab, O., *Some techniques for agile visual tracking and SLAM*. 2008.

47. Gluckman, J.N., S.K.; Dept. of Comput. Sci., Columbia Univ., New York, NY, *Ego-motion and omnidirectional cameras*. Computer Vision, 1998. Sixth International Conference, IEEE: Bombay , India. p. 999 - 1005.

48. Gordon, N.J.S., D.J.; Smith, A.F.M.; Defence Res. Agency, Farnborough *Novel approach to nonlinear/non-Gaussian Bayesian state estimation.* Radar and Signal Processing, IEE Proceedings, 1993. **140**(2): p. 107 - 113.

49. Hager, G.D.D., M.; Stewart, C.V.; Johns Hopkins Univ., Baltimore, MD, USA, *Multiple kernel tracking with SSD*, in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference* 2004. p. I-790 - I-797.

50. Hai Tao; Sawhney, H.S.K., R.; Sarnoff Corp., Princeton, NJ *Dynamic layer representation with applications to tracking*, in *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference*. 2000: Hilton Head Island, SC , USA p. 134 - 141.

51. Hald, A., *History of Probability and Statistics and Their Applications before 1750 (Wiley Series in Probability and Statistics)*. 1990: Wiley Interscience.

52. Hassanpour, E.N.S.S.H., *Edge Detection Techniques: Evaluations and Comparisons.* Applied Mathematical Sciences, 2008. **2**(31): p. 1507 - 1520.

53. Haug, A.J., *A Tutorial on Bayesian Estimation and Tracking Techniques Applicable to Nonlinear and Non-Gaussian Processes.* The MITRE Corporation, 2005.

54. Heiko Helble, S.C., *OATS: Oxford Aerial Tracking System*, in *Robotics and Autonomous Systems*. 2007. p. 661-666.

55. Hertzberg, L.P.a.S.F.a.J., *Robust localization using context in omnidirectional imaging*, In Proc. of the International Conference on Robotics & Automation (ICRA). p. 2072--2077.

56. Hildreth, D.M.a.E., *Theory of Edge Detection.* Royal Society, 1980. **207**: p. 187–217.

57. Hofhauser, A., Steger, C., and Navab, N, *Edge-Based Template Matching and Tracking for Perspectively Distorted Planar Objects*, in *ISVC '08 Proceedings of the 4th International Symposium on Advances in Visual Computing*. 2008: Springer-Verlag Berlin, Heidelberg.

58. Honggab Kim; Romberg, J.W., W.; Sch. of Electr. & Comput. Eng., Georgia Inst. of Technol., Atlanta, GA, USA *Multi-camera tracking on a graph using Markov chain Monte Carlo*, in *Distributed Smart Cameras, 2009. ICDSC 2009. Third ACM/IEEE International Conference*. 2009. p. 1 - 8.

59. Isard, M., *Visual Motion Analysis by Probabilistic Propagation of Conditional Density*. 1998, Oxford University.

60. Isard, M.B., A.; Oxford Univ, *A mixed-state Condensation tracker with automatic model-switching*, in *Computer Vision, 1998. Sixth International Conference* 1998: Bombay , India. p. 107 - 112.

61. James Malcolm , Y.R., Allen Tannenbaum; School of Electrical and Computer Engineering, *Multi-object tracking through clutter using graph cuts*, in *Non-Rigid Registration and Tracking Through Learning (ICCV)*. 2007.

62. Jianbo Shi; Tomasi, C.D.o.C.S., Cornell Univ., Ithaca, NY *Good features to track*, in *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94., 1994 IEEE Computer Society Conference*. 1994 Seattle, WA , USA. p. 593 - 600.

63. Junqiu Wang; Yasushi Yagi; Inst. of Sci. & Ind. Res., O.U., Ibaraki *Switching local and covariance matching for efficient object tracking*, in *Pattern Recognition, 2008. ICPR 2008. 19th International Conference*. 2008, IEEE: Tampa, FL. p. 1-4.

64. Kailath, T. and C.a.S.R.I. Stanford Univ., Menlo Park, CA, *The Divergence and Bhattacharyya Distance Measures in Signal Selection*, in *Communication Technology, IEEE Transactions*. 1967. p. 52 - 60.

65. Kamath, S.-c.S.C.a.C., *Robust techniques for background subtraction in urban traffic video*, in *Visual Communications and Image Processing 2004* 2004: Sethuraman Panchanathan, Bhaskaran Vasudev, San Jose, CA, USA.

66. Kanade, B.D.L.a.T., *An Iterative Image Registration Technique with an Application to Stereo Vision*. 1981. p. 674-679.

67. Karlsson, R.G., F.; Dept. of Electr. Eng., Linkoping Univ., Sweden *Monte Carlo data association for multiple target tracking*, in *Target Tracking: Algorithms and Applications (Ref. No. 2001/174), IEE*. 2001. p. 13/1 - 13/5.

68. Kiam Choo; Fleet, D.J.D.o.C.S., Toronto Univ., Ont. , *People tracking using hybrid Monte Carlo filtering*, in *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference*. 2001: Vancouver, BC , Canada. p. 321 - 328.

69. Kim, Y.M., *Object Tracking in a Video Sequence*.

70. King Yuen Wong; Spetsakis, M.E., *Motion Segmentation by EM Clustering of Good Features*, in *Computer Vision and Pattern Recognition Workshop, 2004. CVPRW '04. Conference*. 27-02 June 2004: York University, Canada. p. 166 - 166.

71. Kobilarov, M.S., G.; Hyams, J.; Batavia, P.; Robotic Embedded Syst. Lab., Southern California Univ., Los Angeles, CA, *People tracking and following with mobile robot using an omnidirectional camera and a laser*, Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE p. 557 - 562

72. Labrosse, F., *Short and long-range visual navigation usingwarped panoramic images*. 2007. p. 675-684.

73. Larry Davis , V.P., Ramani Duraiswami, *Tracking Humans From a Moving Platform*, in *15th Int. Conf. on Pattern Recognition*. 2000. p. 171-178.

74. Leonard, J.J.D.-w., H.F., *Simultaneous map building and localization for an autonomous mobile robot*, in *Intelligence for Mechanical Systems, Proceedings IROS'91. IEEE/RSJ International Workshop*. 1991. p. 1442–1447.

75. Liang Wang , T.T., Weiming Hu; National Labroratory of Pattern Recognition, *Face Tracking Using Motion-Guided Dynamic Template Matching*, in *The 5th Asian Conference on Computer Vision*. 2002: Melbourne, Australia.

76. Liang-Guo Zhang, X.C., Chunli Wang, and Wen Gao, *Robust Automatic Tracking of Skin-Colored Objects with Level Set based Occlusion Handling*.

77. Lopez-Garcia, F.D.d.I.d.S.y.C., Univ. Politec. de Valencia, Valencia, Spain *SIFT features for object recognition and tracking within the IVSEE system*, in *Pattern Recognition, 2008. ICPR 2008. 19th International Conference*. 2008: Tampa, FL. p. 1 - 4.

78. Lowe, D.G.D.o.C.S., British Columbia Univ., Vancouver, BC *Object recognition from local scale-invariant features*, in *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference*. 1999: Kerkyra , Greece. p. 1150 - 1157.

79. Lucas, B.D., *Generalized image matching by the method of differences*. 1984, Carnegie Mellon University Pittsburgh, PA, USA.

80. MacCormick, J.B., A.; Oxford Univ., *A probabilistic exclusion principle for tracking multiple objects*, in *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference*. 1999: Kerkyra , Greece. p. 572 - 578.

81. Malcolm, J.G.R., Yogesh ; Tannenbaum, Allen R., *Tracking Through Clutter Using Graph Cuts*. 2007, Georgia Institute of Technology.

82. Michael J Black, A.D.J., *EigenTracking: Robust Matching and Tracking of Articulated Objects Using a View-Based Representation.* International Journal of Computer Vision, 1998. **26**(1): p. 63-84.

83. Michael Kass , A.W., Demetri Terzopoulos, *Snakes: Active contour models.* INTERNATIONAL JOURNAL OF COMPUTER VISION, 1988. **1**: p. 321--331.

84. Moore, T.B., *Implementation of Object Tracking Using SIFT Keypoints.* 2006.

85. Nan Jiang; Wenyu Liu; Ying Wu; Dept. of Electron. & Inf. Eng., H.U.o.S.T., Wuhan, China *Learning Adaptive Metric for Robust Visual Tracking*, in *Image Processing, IEEE Transactions*. 2011, IEEE Signal Processing Society. p. 2288 - 2300.

86. Natarajan, B., *Tracking of objects in spatiotemporal volume by graph-cuts*. 2009.

87. Niall Winters, J.e.S.-V., *Information Sampling for Optimal Image Data Selection.* 2002: p. 145-159.

88. Nicole M. Artner, A.I., Walter G. Kropatsch, *Coarse-to-Fine Tracking of Articulated Objects Using a Hierarchical Spring System*. 2009: CAIP.

89. Ning Xu , N.A., Ravi Bansal, *Object segmentation using graph cuts based active contours*. 2007.

90. nyi, L.C.T.s.S., *Motion Segmentation and Tracking with Edge Relaxation and Optimization using Fully Parallel Methods in the Cellular Nonlinear Network Architecture.* Real-Time Imaging, 2001. **7**: p. 77-95.

91. P. Smith , T.D., R. Cipolla; Department of Engineering University of Cambridge, *Edge Tracking for Motion Segmentation and Depth Ordering*. 1999.

92. R. E. KALMAN; Research Institute for Advanced Study, B., Md., *A New Approach to Linear Filtering and Prediction Problems.* Transactions of the ASME-Journal of Basic Engineering. **82**: p. 35-45.

93. Ramesh Jain, R., Brian G. Schunck, *Machine Vision*. 1995: McGraw-Hill, Inc.

94. Roberts, C.K.A.G.S.N.G.M., *Salient points for tracking moving objects in video*, in *Image and Video Communications and Processing*. 2005. p. 442-453.

95. Ronfard, R., *Region-based strategies for active contour models*, in *Int. J. Comput. Vision,* . 1994. p. 229-251.

96. Rubin, A.P.D.a.N.M.L.a.D.B., *Maximum likelihood from incomplete data via the EM algorithm.* JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B, 1977. **39**(1): p. 1-38.

97. S.S.Mohith, *Real time interactive Object tracking*, in *Institute of Science and Technology*. 1998, University of Manchester: Manchester.

98. Seelen, U.H.a.T.K.a.C.T.a.M.W.a.W.V., *An Image Processing System for Driver Assistance*. 1998, Image and Vision Computing. p. 367-376.

99. Sethian, J.A., *Level Set Techniques for Tracking Interfaces; Fast Algorithms, Multiple Regions, Grid Generation, and Shape/Character Recognition.*

100. Shimshoni, A.A.a.E.R.a.I., *Robust Fragments-based Tracking using the Integral Histogram*, in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*. 2006. p. 798-805.

101. Slawo Wesolkowski and Ed Jernigan; University of Waterloo, C., *Color Edge Detection in RGB Using Jointly Euclidean Distance and Vector Angle*, in *Vision Interface*. 1999: Canada.

102. Smith, S.M., *Reviews of Optic Flow, Motion Segmentation, Edge finding and Corner Finding*, in *Oxford Centre for Functional Magnetic Resonance Imaging of the Brain (FMRIB)*. 1992, Oxford University.

103. Stauffer, C.G., W.E.L.; Artificial Intelligence Lab., MIT, Cambridge, MA, *Adaptive background mixture models for real-time tracking*, in *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference*. 1999: Fort Collins, CO , USA.

104. Stauffer, C.G., W.E.L.; Artificial Intelligence Lab., MIT, Cambridge, MA *Learning patterns of activity using real-time tracking*, in *Pattern Analysis and Machine Intelligence, IEEE Transactions*. 2000. p. 747 - 757

105. Stephen C Pratt, S.E.B., Nigel R Franks; Centre for Mathematical Biology, and Department of Biology and Biochemistry, University of Bath, *The use of edges in visual navigation by the ant Leptothorax albipennis*. 2001: Blackwell Wissenschafts-Verlag, Berlin. p. 1125-1136.

106. Sukhatme, B.J.a.G.S., *Detecting moving objects using a single camera on a mobile robot in an outdoor environment.* in International Conference on Intelligent Autonomous Systems, 2004: p. 980--987.

107. Ting Yu; Cha Zhang; Michael Cohen; Yong Rui; Ying Wu; GE Global Research, N., *Monocular Video Foreground/Background Segmentation by Tracking Spatial-Color Gaussian Mixture Models*, in *Motion and Video Computing, 2007. WMVC '07. IEEE Workshop*. 2007: Austin, TX, USA. p. 5 - 5.

108. Tristan Mitchell and Frédéric Labrosse; Department of Computer Science, U.o.W., Aberystwyth, Ceredigion, SY23 3DB, *Visual homing: a purely appearance-based approach*, in *University of Essex*, Proceeding of Towards Autonomous Robotic Systems. p. 101--108.

109. Vermaak, J.G., S.J.; Perez, P.; Dept. of Eng., Cambridge Univ., UK, *Monte Carlo filtering for multi target tracking and data association*, in *Aerospace and Electronic Systems, IEEE Transactions*. 2005. p. 309 - 332.

110. Vicent Caselles , R.K., Guillermo Sapiro, *Geodesic Active Contours.* International Journal of Computer Vision, 1995. **22**: p. 61-79.

111. Visser, J.S.a.P.V.R.a.A., *Panoramic Localization in the 4-Legged League Removing the dependence on artificial landmarks*, In RoboCup(2006). p. 387-394.

112. Walker, H.M., *Studies in the History of Statistical Method: Special Reference to Certain Educational Problems*. 1975: Ayer Co Pub; Facsimile of 1929 ed edition.

113. Priti P.Kuralkar, Prof. V.T.Gaikwad., *Human Object Tracking using Background Subtraction and Shadow Removal Techniques*. 2012: International Journal of Advanced Research in Computer Science and Software Engineering. Volume 2, Issue 3.

114. Williams, R.G.A.L.R., *Multiple Target Tracking with Lazy Background Subtraction and Connected Components Analysis*, in *Springer-Verlag New York, Inc. Secaucus*. 2005, Machine Vision and Applications: NJ, USA.

115. Wilson, J. *Wallpaper,* Available at *http://wilstar.com/wallpaper*. (Accessed 10th April 2011)

116. Wren, C.A., A.; Darrell, T.; Pentland, A.; Media Lab., MIT, Cambridge, MA, *Pfinder: real-time tracking of the human body*, in *Automatic Face and Gesture Recognition, 1996., Proceedings of the Second International Conference*. 1996: Killington, VT , USA. p. 51 - 56.

117. Xiaoqin Zhang; Weiming Hu; Maybank, S.X.L.I.o.A., Beijing *Graph Based Discriminative Learning for Robust and Efficient Object Tracking*, in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference*. 2007: Rio de Janeiro. p. 1 - 8.

118. Xiaoyu Wang, G.H., and Tony X. Han, *Discriminative tracking by metric learning*, in *ECCV'10 Proceedings of the 11th European conference on computer vision conference on Computer vision: Part III*. 2010.

119. Xu, N.A., N.; Beckman Inst. for Adv. Sci. & Technol., Illinois Univ., *Object contour tracking using graph cuts based active contours*, in *Image Processing. 2002. Proceedings. 2002 International Conference*. 2002: Urbana, IL, USA. p. III-277 - III-280.

120. Xue Mei, H.L., *Robust Visual Tracking using L1 Minimization*. 2009: ICCV.

121. Yi Wu; Bo Wu; Jia Liu; Hanqing Lu; Inst. of Autom., C.A.o.S., Beijing, China, *Probabilistic tracking on Riemannian manifolds*, in *Pattern Recognition, 2008. ICPR 2008. 19th International Conference*. 2008: Tampa, FL. p. 1 - 4.

122. Ying Wu; Jialue Fan; Northwestern Univ., E., IL, USA *Contextual Flow*, in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference*. 2009: Miami, FL. p. 33 - 40.

123. Yiwei Wang; Doherty, J.F.V.D., R.E.; Dept. of Electr. Eng., Pennsylvania State Univ., University Park, PA, *Moving object tracking in video*, in *Applied Imagery Pattern Recognition Workshop, 2000. Proceedings. 29th*. 2000 Washington, DC , USA. p. 95 - 101.

124. Yizong Cheng; Dept. of Electr. & Comput. Eng., C.U., OH *Mean shift, mode seeking, and clustering*, in *Pattern Analysis and Machine Intelligence, IEEE Transactions*. 1995. p. 790 - 799.

125. Yokoyama, M.P., T.; Div. of Adv. LSI Design & Dev., Sony Corp.,, *A contour-based moving object detection and tracking*, in *Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005. 2nd Joint IEEE International Workshop*. 2005: Tokyo, Japan. p. 271 - 276.

126. Younse, P.J.B., T. F; Dept of Agricultural and Biological Engineering, University of Florida, Gainesville, FL, USA, *Greenhouse Robot Navigation Using KLT Feature Tracking for Visual Odometry*. 2007, International Commission of Agricultural Engineering.

127. Yunqiang Chen; Huang, T.S.Y.R.B.I., Illinois Univ., Urbana, IL, *Optimal radial contour tracking by dynamic programming*, in *Image Processing, 2001. Proceedings. 2001 International Conference*. 2001: Thessaloniki , Greece. p. 626 - 629.

128. Zhenjun Han, Q.Y., Jianbin Jiao; Graduate University of Chinese Academy of Sciences, 100049 Beijing, PR China, *Combined feature evaluation for adaptive visual object tracking.* Computer Vision and Image Understanding, 2010. **115**(1): p. 69-80.

129. *Fast Fourier Transform(FFT): How to implement the FFT algorithm,* Available at *http://www.codeproject.com/KB/recipes/howtofft.aspx.* (Accessed 14th May 2011)

130. *KLT: An Implementation of the Kanade-Lucas-Tomasi Feature Tracker, http://www.ces.clemson.edu/~stb/klt/.*

131. *Image Scanning Techniques,*
Available at *http://www.axis.com/products/video/camera/progressive_scan.htm,* (Accessed 15th June 2011)

132. Danny Pascale; A Review of RGB Color Spaces … From xyY to R'G'B', 2002-2003.

133. Lance Fortnow, *Ravi Kannan Honored for Advances in Algorithmic Technique.* ACM. *2011.*

134. "*Proceedings of the Winter Meeting of the Optical Society of America", JOSA, 1994.*, Vol. 40, Issue 4, pp. 254-254.

134. Donald E. Knuth, "*The Art of Computer Programming: Seminumerical Algorithms",* Vol 2, 3rd Ed, 1997, Addison-Wesley, ISBN 0201896842.

135. Kruskal and Wallis, *"Use of ranks in one-criterion variance analysis",* Journal of the American Statistical Association, *1952,* **47** (260): 583–621.

136. "*Minitab 16 for Statistics - Power to Understand Your Data",* Available at *http://www.minitab.com* (Accessed 17th August 2011).