CONSTRUCTING PANORAMIC SCENES FROM AERIAL VIDEOS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ELİF ERDEM

IN PARTIAL FULLFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

DECEMBER 2007

Approval of the thesis:

CONSTRUCTING PANAROMIC SCENES FROM AERIAL VIDEOS

submitted by **ELİF ERDEM** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen _____
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. İsmet Erkmen _____
Head of Department, **Electrical and Electronics Engineering**

Assoc. Prof. Dr. Gözde Bozdağı Akar _____
Supervisor, **Electrical and Electronics Engineering Dept., METU**

Examining Committee Members:

Prof. Dr. Uğur Halıcı _____
Electrical and Electronics Engineering Dept., METU

Assoc. Prof. Dr. Gözde Bozdağı Akar _____
Electrical and Electronics Engineering Dept., METU

Assoc. Prof. Dr. Tolga Çiloğlu _____
Electrical and Electronics Engineering Dept., METU

Assist. Prof. Dr. İlkay Ulusoy _____
Electrical and Electronics Engineering Dept., METU

Dr. Hüseyin Yavuz _____
ASELSAN Inc.

Date: ____04.12.2007_____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: Elif ERDEM

Signature :

ABSTRACT

CONSTRUCTING PANORAMIC SCENES FROM AERIAL VIDEOS

ERDEM, Elif

M.S., Department of Electrical and Electronics Engineering

Supervisor: Assoc. Prof. Dr. Gözde Bozdağı Akar

December 2007, (124) pages

In this thesis, we address the problem of panoramic scene construction in which a single image covering the entire visible area of the scene is constructed from an aerial image video.

In the literature, there are several algorithms developed for construction of panoramic scene of a video sequence. These algorithms can be categorized as feature based and featureless algorithms. In this thesis, we concentrate on the feature based algorithms and comparison of these algorithms is performed for aerial videos. The comparison is performed on video sequences captured by non-stationary cameras, whose optical axis does not have to be the same. In addition, the matching and tracking performances of the algorithms are separately

analyzed, their advantages-disadvantages are presented and several modifications are proposed.

Keywords: Image Mosaicing, Homography, Phase Correlation, RANSAC, Least Median of Squares (LMedS), Harris, Minimum Eigenvalue Method, SIFT.

ÖZ

HAVADAN ÇEKİLMİŞ YER VİDEOLARINDAN PANORAMİK GÖRÜNTÜ OLUŞTURULMASI

ERDEM, Elif

Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü

Tez Yöneticisi: Assoc. Prof. Dr. Gözde Bozdağı Akar

Aralık 2007, (124) sayfa

Bu tez, havadan çekilmiş yeryüzü videolarından panoramik görüntü oluşturulması konusunu içeren bir çalışmadır.

Literatürde panoramik görüntü oluşturulmasında kullanılan birçok algoritma bulunmaktadır. Bu algoritmalar özellik tabanlı ve benzerlik tabanlı olarak gruplandırılabilir. Bu çalışmada, özellik tabanlı algoritmaların karşılaştırılması üzerinde yoğunlaşılmış ve havadan çekilmiş yeryüzü videolar için karşılaştırma yapılmıştır. Karşılaştırma sırasında kullanılan videolar hareketli kameralardan elde edilmiş olup optik odağı sabit olmayan kameralar kullanılmıştır. Bu çalışmada, özellik tabanlı algoritmaların eşleme ve takip performansları analiz

edilmekte ve avantaj ve dezavantajları belirtilip, yapılan iyileştirme çalışmaları sunulmaktadır.

Anahtar Kelimeler: Panoramik görüntü, homografi, faz ilişkilendirme, RANSAC, LMedS, Harris, Minimum Özdeğer Metodu, SIFT.

To those who contribute to this thesis…

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude and appreciation to my supervisor Assoc. Prof. Dr. Gözde Bozdağı Akar for her guidance, valuable suggestions and deepest knowledge and Anıl Aksay for his support at all levels of this study.

I would also like to thank to my colleagues for their assistance and patient. And of course to my company ASELSAN and Mr. Hüseyin YAVUZ for his large perspective, guidance and support at all level of this study.

I would like to also express my thanks for their friendship to my cousin Mustafa, his wife Pınar and I also want to thank to Sinan for his friendship.

I would like to thank my Mom, Dad and brother Emre for their love. I feel their support always with me throughout my life and I suppose this is the source of all achievements of my life.

And of course I would like to express my greatest thanks to Melih for his love, invaluable support, encouragement and patience. This thesis is dedicated to them.

TABLE OF CONTENTS

ABSTRACT iv

ÖZ. vi

ACKNOWLEDGEMENTS ix

TABLE OF CONTENTS x

LIST OF FIGURES xii

LIST OF TABLES xv

LIST OF ABBREVIATIONS xvi

1 INTRODUCTION 1

1.1 Motivation and Objective 7

1.2 Organization 8

2 FEATURELESS MOSAICING ALGORITHMS 9

2.1 Phase Correlation 10

2.2 Pixel Based Algorithms 20

3 FEATURE BASED MOSAICING ALGORITHMS 21

3.1 Feature Extraction 26

3.1.1 Feature Extraction Algorithms 26

3.1.1.1 Minimum Eigenvalue Method 26

3.1.1.2 Harris Corner Detector 34

3.1.1.3 SIFT Keypoint Extraction by Difference of Gaussian Method 39
3.2 Feature Tracking 43
3.2.1 Pyramidal Iterative Lucas and Kanade 44
3.2.2 SIFT Matching Algorithm 49
3.3 Transformation Matrix and Mosaic Image Construction 50
3.3.1 Transformation Matrix Model 51
3.3.1.1 Six -6- Parameter Affine Transformation 51
3.3.1.2 Eight -8- Parameter Homography Transformation 52
3.3.2 Transformation Matrix calculation Algorithms 53
3.3.2.1 Least Squares 53
3.3.2.2 RANSAC 54
3.3.2.3 LMedS 56
4 RESULTS AND COMPARISON OF MOSAICING ALGORITHMS 59
4.1 Mosaicing Algorithm 1 63
4.2 Mosaicing Algorithm 2 81
4.3 Mosaicing Algorithm 3 91
4.4 Mosaicing Algorithm 4 100
4.5 Comparison of the Algorithms 114
5 SUMMARY AND CONCLUSIONS 119
5.1 Future Work 121
6 REFERENCES 123

LIST OF FIGURES

Figure 1: An 1851 panoramic showing San Francisco from Rincon Hill by photographer Martin Behrmanx. (www.wikipedia.com) 1

Figure 2: Block Schema of mosaicing procedure 3

Figure 3: Phase Correlation Result for white square on black background 13

Figure 4: Input frames from zeppelin video for Phase Correlation Algorithm (a) 15th frame (b) 200th frame (Example-1) 14

Figure 5: Input frames from zeppelin video for Phase Correlation Algorithm (a) 15th frame (b) 32nd frame (Example-2)..... 16

Figure 6: Input frames from UAV video for Phase Correlation Algorithm (a) 15th frame (b) 32nd frame (Example-3)..... 18

Figure 7: Flow chart of a mosaicing algorithm..... 23

Figure 8: Minimum Eigenvalue Feature Selection Algorithm 33

Figure 9: Amplitude of the Response Function, |R|..... 37

Figure 10: Harris Feature Detection Algorithm..... 38

Figure 11: Pseudo-code of the Pyramidal KLT algorithm 48

Figure 12: Original full frame images of (a)"wall" (c) "amasra" videos. Transformed Original Images with respect to first frames of (b)"wall"

(d)”amasra” videos. Pixel intensity differences are calculated from (b) and (d) images. 61
Figure 13: Original Image 63
Figure 14: Flowchart of Mosaic Algorithm 1 65
Figure 15: Features of Harris Method in real videos (a) zeppelin (b) uav 68
Figure 16: Features of Harris Method in synthetic videos (a) wall (b) amasra 69
Figure 17: Cropped Image of Figure 13 72
Figure 18: Track error of KLT in x and y directions 72
Figure 19: Homography Parameter Error of Least Squares 75
Figure 20: Mosaic Results of Mosaicing Algorithm 1 for real videos (a) zeppelin (b) uav 77
Figure 21: Mosaic Results of Mosaicing Algorithm 1 for synthetic videos (a) wall (b) amasra 78
Figure 22: Pixel difference of mosaic image and original image in (a)wall (b)amasra videos 80
Figure 23: Homography Error for RANSAC 84
Figure 24: Flowchart of Mosaic Algorithm 2 85
Figure 25: RANSAC Implementation 86
Figure 26: Mosaic Results of Mosaic Algorithm 2 for synthetic videos (a)zeppelin (b)uav 87
Figure 27: Mosaic Results of Mosaic Algorithm 2 for synthetic videos (a) wall (b) amasra videos 88
Figure 28: Pixel-wise frame difference for (a) wall (b) amasra mosaic 90

Figure 29: Homography Error for LMedS Algorithm 93

Figure 30: Flowchart of Mosaicing algorithm 3 94

Figure 31: LMedS Implementation..... 95

Figure 32: Mosaic Results of Mosaic Algorithm 3 for real videos (a) zeppelin (b) uav videos 96

Figure 33: Mosaic Results of Mosaic Algorithm 3 for synthetic videos (a) wall (b) amasra videos 97

Figure 34: Pixel-wise frame difference of MA3 for (a) wall (b) amasra mosaics..... 99

Figure 35: Flowchart of Mosaicing Algorithm 4 102

Figure 36: Keypoints extracted by SIFT algorithm(DoG) for real videos (a) zeppelin (b) uav 104

Figure 37: Keypoints extracted by SIFT algorithm (DoG) for synthetic videos (a) wall (b) amasra 105

Figure 38: Distance between the matched points 108

Figure 39: Detailed first portion of Figure 38..... 109

Figure 40: Mosaic Results of Mosaic Algorithm 4 for real videos (a) zeppelin (b) uav videos 111

Figure 41: Mosaic Results of Mosaic Algorithm 4 for real videos (a) zeppelin (b) uav videos 112

Figure 42: Pixel-wise frame difference for (a) amasra (b) wall videos..... 113

LIST OF TABLES

Table 1: Displacement table for Example 2 17

Table 2: Displacement table for Example 3 19

Table 3: Steps of Implemented Algorithms..... 60

Table 4: Minimum distance and eigenvalue parameters for the videos..... 67

Table 5: Parameters of KLT of the videos 71

Table 6: Number of features and thresholds..... 74

Table 7: SNR for the synthetic videos..... 76

Table 8: SNR values for MA3..... 89

Table 9: SNR values of synthetically generated videos for MA3 98

Table 10: SIFT Keypoint Determination Parameters..... 103

Table 11: Parameter of SIFT matching..... 107

Table 12: SNR values for SIFT algorithm..... 110

Table 13: SNR Values of the algorithms 115

Table 14: Time complexity of the algorithms 117

Table 15: Spent time for algorithms for two different frames 117

Table 16: Spent time for algorithms for two different frames_ Consecutive Homography..... 118

LIST OF ABBREVIATIONS

Det	: Determinant
DoG	: Difference of Gaussian
Eq	: Equation
FFT	: Fast Fourier Transform
SIFT	: Scale Invariant Feature Transform
KLT	: Kanade-Lucas Tracker
LMedS	: Least Median of Squares
LS	: Least Squares
MA	: Mosaicing Algorithm
Min	: Minimum
MPEG4	: Moving Picture Experts Group 4
RANSAC	: Random Sample Consensus
SSD	: Sum of Squared Difference
SNR	: Signal to Noise Ratio
Tr	: Trace
UAV	: Unmanned Aerial Vehicle

CHAPTER 1

INTRODUCTION

When Mr. Martin Behrmanx produced probably the first panoramic scene in 1843, possibly he could not imagine that this scene can be constructed in an automatic manner. To construct such a scene like the one in Figure 1, careful attention and manual processes were required.

Figure 1: An 1851 panoramic showing San Francisco from Rincon Hill by photographer Martin Behrmanx. (www.wikipedia.com)

In the age of information, today it is easier to generate a panoramic view of given still images when compared with 1850's. Now, the digital cameras and computer programs can automatically generate panoramic view using only the translation

information between the frames. Naturally, these methods are all offline processes and perspective changes are not taken into account. On the other hand, it is much more difficult to obtain panoramic scenes of videos than still images. This is due to the fact that for video case, camera motion should be modeled and perspective changes in the video sequence should be evaluated correctly.

Mosaic image of video sequences is useful in applications where the complete information in the video sequence is necessary. Besides, if there is lack of storage capacity or bandwidth constraint, mosaicing becomes a necessary procedure. For instance, in MPEG4 mosaic image of background is constructed for efficient compression. Additionally, in applications where stabilization of the image is required for stable background construction, mosaicing is widely used.

In the literature, there are several approaches to handle the mosaicing problem. In mosaic image construction, the aim is to find the transformation between the image pair and warp one of the images with respect to the other one. Then these images are combined and the mosaic image is constructed. The block schema of mosaicing process is given in Figure 2.

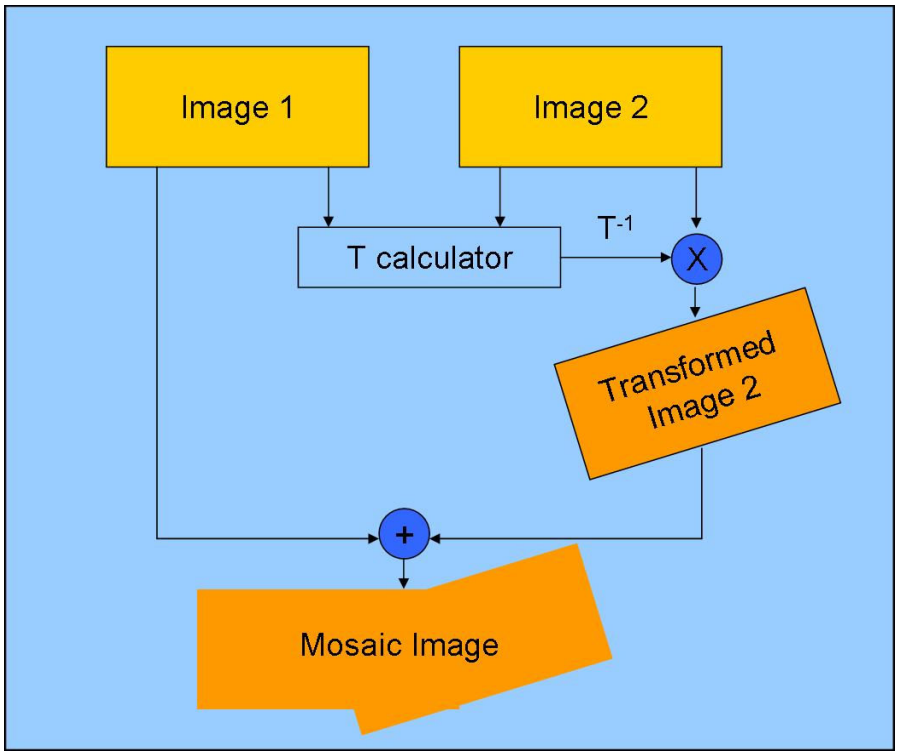


Figure 2: Block Schema of mosaicing procedure

The mosaicing approaches can be classified into two: featureless and feature-based methods where a feature corresponds to the distinctive interest point in an image frame. In the literature, [1], [2], [12] and [13] utilize featureless methods whereas [11] and [15] exploit feature-based algorithms for mosaicing. System [1] uses a three step approach to measure image motion from low-quality images. Firstly, phase-correlation is used to recover inter-frame global motion for approximate aligning. Then, refinement of the inter-frame transformation is performed by estimating the optical flow between the compensated images and fitting a parametric motion model to this flow. Finally, the Sum of Squared Difference (SSD) between the images is minimized where the SSD function is

formed by careful selection of the image regions using the edge structure. This algorithm estimates the transformation parameters accurately, and produces accurate mosaics for different videos. This has advantages in applications where foreground and translational motions exist. For applications in which rotation dominates translation as Zeppelin application, model fitting increases the time complexity.

In [2], unknown transformation parameters are directly recovered from measurable image quantities as intensity of each pixel in the image. In this method, a brightness constancy constraint is assumed for every pixel of the image and coarse-to-fine processing is utilized using iterative refinement within a multi-resolution pyramid. All processing is performed in sub pixel accuracy. To find out the transformation matrix, the algorithm tries to fit a homography model. This approach has the disadvantage of high computational complexity. [2] also shows that direct methods are capable of recovering misalignments of up to 10-15 % of the image size whereas an initial estimate is required for larger misalignments.

[13] is another intensity based algorithm where a Laplacian pyramid is constructed from both of the input images and a model is fitted in coarse-fine manner. The algorithm exploits quadratic image motion model where SSD is the match measure. This method deals with all pixel values so time complexity and noise sensitivity is large.

[15] uses a feature based algorithm where features are extracted before structure and motion recovery. Then, these features are used to compute image matching relations. The algorithm automatically estimates a homography between the features of two images using Random Sample Consensus (RANSAC) algorithm. Besides, it computes a maximum likelihood estimate of homographies and points over all frames. This application is very similar to one of the algorithms implemented in this study except for the maximum likelihood estimation. This method can be applied if there is foreground motion in the video.

Another feature based algorithm is presented by [11] in which image alignment is performed by fitting a global parametric motion model to sparse optic flow. The Kanade-Lucas-Tomasi (KLT) feature tracker is used to match corner features between adjacent pairs of video frames to obtain a sparse estimate of the optic flow field. For each corner feature, the method solves for a sub pixel translational displacement vector that minimizes the sum of squared intensity differences between an image patch centered at the corner and a patch in the next frame centered at the estimated translated position. A six parameter affine motion model is fit to the observed displacement vectors between two frames to approximate the global flow field induced by camera motion and a rigid ground plane. RANSAC algorithm is used to robustly estimate the affine parameters from the observed displacement vectors. The benefit of using a robust procedure such as RANSAC is that the final least squares estimate is not contaminated by erroneous displacement vectors, such as points on moving vehicles in the scene, and scene

points with large parallax. This method can only produce good results when the displacement between frames can be modeled by affine motion

[8] proposes a mixture of these algorithms where a feature-based method is used to initialize the registration, i.e., to compute camera motion and layer segmentation. Then registration fine tuning is performed via a featureless method. This method uses both feature-based and featureless methods which increase the complexity. However, in order to decrease the time complexity, it is efficient to use feature points/characteristics for fine tuning.

Featureless methods have very large complexity since they deal with every pixel in the image. On the other hand, they solve two problems simultaneously: the motion of the camera and the correspondence of every pixel. However, they process less informative portion of the image and are affected from the image brightness constraint and noise more than feature based algorithms

By contrast, feature based approaches involve a strategy of concentrating computation on areas of the image where it is possible to get good correspondence, so that an initial estimate of geometry is constructed. This geometry is then used to guide correspondence in regions of the image where there is less information. So the process is focused on more informative portion of the image and complexity of the algorithm decreases effectively.

1.1 Motivation and Objective

In this study, the problem of mosaic image construction is addressed and different algorithms in the literature are implemented and compared. The comparison criteria of these algorithms are visual performance (SNR), computational load and time complexity.

As explained above, automatic construction of mosaic from image sequences can be realized using different algorithms. The appropriate method should be chosen according to video sequences and requirements of the system. In this thesis, we focus on aerial videos taken from a high altitude camera so depth information is ignored in order to decrease time complexity. Additionally, refinement procedures for the algorithms are adjusted accordingly for aerial video types.

In this study, both synthetic and real videos are utilized. For real videos, two videos with different characteristics are exploited. One of the videos is “zeppelin” video in which camera is connected loosely to the balloon with wires allowing camera’s optical (focal) axis to change. The situation is that the balloon is connected to the ground from three different contact points so it can move in the air and the camera moves with this balloon. The altitude of the camera is about 60m. The video has a resolution of 288x384. The second video is “uav” video. This is the video utilized in [1]. The camera is tightly connected to an unmanned air vehicle (UAV). As the UAV moves, camera connected to it takes the video in which there is a moving object. The color video has a resolution of 230x310.

In addition to real videos, two synthetic videos are used in comparison of the algorithms. The first synthetic video is obtained by grabbing small portions of a high resolution Amasra image which is taken from a high altitude. The resolution is 640x480. Since the camera used to grab the frames has no manual focus adjustment, brightness differences present between the frames in the "amasra" video. The last video is synthetic "wall" video. This video is taken from a wall on which there are pictures and paintings. The reason for this is to make resemble the synthetic video to real aerial videos and to prevent depth information to present in the sequences (as in aerial sequences) which makes correct mosaic construction possible. In this work, the results of the algorithms for these four videos are presented and discussed.

1.2 Organization

In Chapter 2, Featureless mosaicing algorithms in the literature are reviewed and some implementation results are given. Third chapter describes theoretical background of feature-based techniques in construction of mosaic images. Fourth chapter depicts comparison of the algorithms for different situations and videos. The results of the implementations and comments about the implemented methods are presented in this chapter. In chapter 5, a brief summary of this study, conclusions about the results obtained and comparison of the methods and furthermore, recommendations for possible future works are given.

CHAPTER 2

FEATURELESS MOSAICING ALGORITHMS

In featureless mosaicing algorithms, the transformation between the frames is calculated using measurable image quantities of every pixel in the image. In this approach, all pixel information is utilized without extracting any informative region. These algorithms differ in the kind of information extracted from the image. In this study two algorithms are mentioned:

1. Phase Correlation : Estimates global motion using frequency information
2. Pixel Based Approach: Estimates transformation using intensity values of each pixel in the image

2.1 Phase Correlation

Phase correlation is a method that use frequency domain map of the signal. This method provides a rough estimate for rigid translational motion between two images.

In this application, FFT phase correlation is used to recover inter-frame translation, so before applying a detailed procedure, images are aimed to be aligned roughly. Phase correlation method gives spatial shift of the signal as the output. This method measures the translation directly from their phases. This algorithm is optimum in the complexity point of view. This is because it does not include any gradient based estimation of the parameters, so it gives information about planar translation in a fast manner.

The phase correlation method is applied in order to find the intra frame planar translation. This means one of the frames has a displacement of (d_1, d_2) with respect to the second frame in x and y directions, respectively.

$$I_{k+1}(n_1, n_2) = I_k(n_1 + d_1, n_2 + d_2) \tag{1}$$

The shift in the spatial domain corresponds to phase change in the spectrum domain. Taking the Fourier Transform of both sides of Equation (1) we have:

$$I_{k+1}(f_1, f_2) = I_k(f_1, f_2)\exp\{-j2\pi(f_1 d_1 + f_2 d_2)\} \tag{2}$$

In phase correlation calculation, the first step is to find cross correlation between the k^{th} and $(k+1)^{th}$ frames as:

$$c_{k,k+1}(n_1, n_2) = I(n_1, n_2, k+1) ** I(-n_1, -n_2, k) \tag{3}$$

where ** stands for 2D convolution. Then the Fourier Transform of Equation (3) is taken to obtain cross power spectrum:

$$C_{k,k+1}(f_1, f_2) = I_{k+1}(f_1, f_2) I_k^{*}(f_1, f_2) \tag{4}$$

where * stands for complex conjugate. In order to eliminate luminance variations, the cross power spectrum is normalized by its magnitude. The Fourier transform of the cross correlation matrix is:

$$\widetilde{C}_{k,k+1} = \frac{I_{k+1}(f_1, f_2) I_k^{*}(f_1, f_2)}{\left| I_{k+1}(f_1, f_2) I_k^{*}(f_1, f_2) \right|} \tag{5}$$

If Equation (4) and Equation (5) are combined and inverse Fourier transform is taken:

$$\widetilde{C}_{k,k+1}(f_1, f_2) = \exp\{-j2\pi(f_1 d_1 + f_2 d_2)\} \tag{6}$$

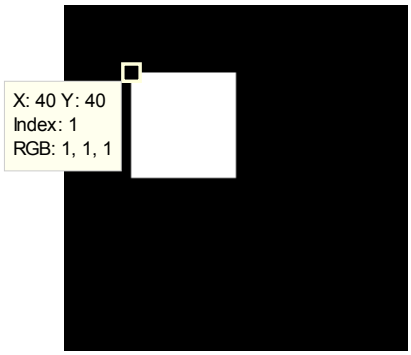
$$\widetilde{c}_{k,k+1}(n_1, n_2) = \delta(n_1 - d_1, n_2 - d_2) \tag{7}$$

This delta function contains the translation information between the frames k^{th} and $(k+1)^{th}$. The planar translation is 2 dimensional so it has magnitude in x and y directions. The displacement between the frames is obtained by applying the

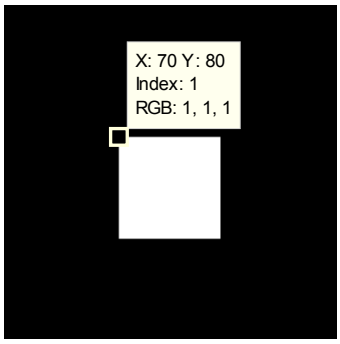
above formulas and the location of the pulse in Equation (7) gives the displacement in x and y directions which are d_1 and d_2, respectively.

Actually this is the ideal case because in practice the motion between frames is not pure translational and focal axes of the cameras do not have to stay unchanged. As a result there also exists rotation. In non pure translational case, more than one pulse exists and the pulse that has the highest magnitude gives the translation information. The phase correlation method gives accurate results if the translational motion between the frames is dominant. If rotation dominates, even if there is translational motion, phase correlation operation gives insufficient/incorrect result.

In order to see phase correlation performance, a size of 40x40 white square on black background is created at location (40,40) in Figure 3a. Then the white square is shifted to the location (70,80) in Figure 3b The shift between the frames is found by phase correlation method and the phase correlation algorithm outputs a peak at location (30,40). This output is shown as the white square in Figure 3c which indicates 30 pixels and 40 pixels shifts in x and y directions respectively. It is observed that phase correlation algorithm works perfectly in this situation.



(a)



(b)

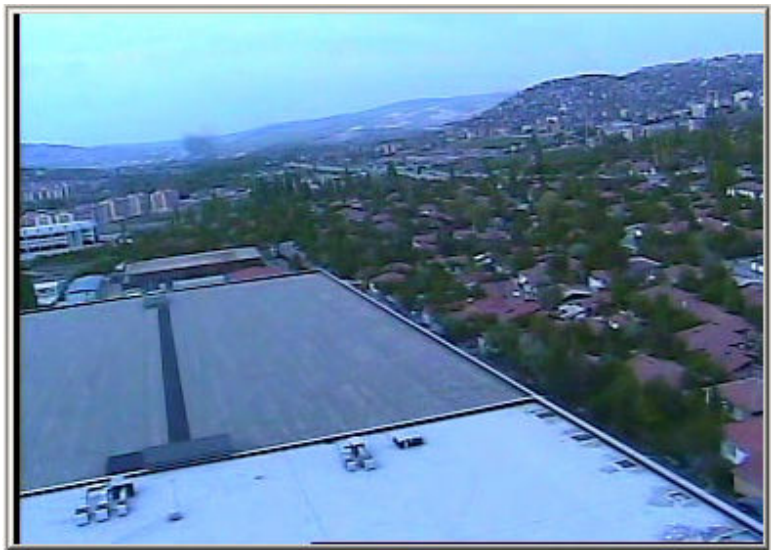
(c) Phase Correlation in spatial domain

Figure 3: Phase Correlation Result for white square on black background

a) White square at (40,40) b) Shifted white square at (70,80)

c) Phase correlation result: Displacement between (a) and (b)

(a)



(b)

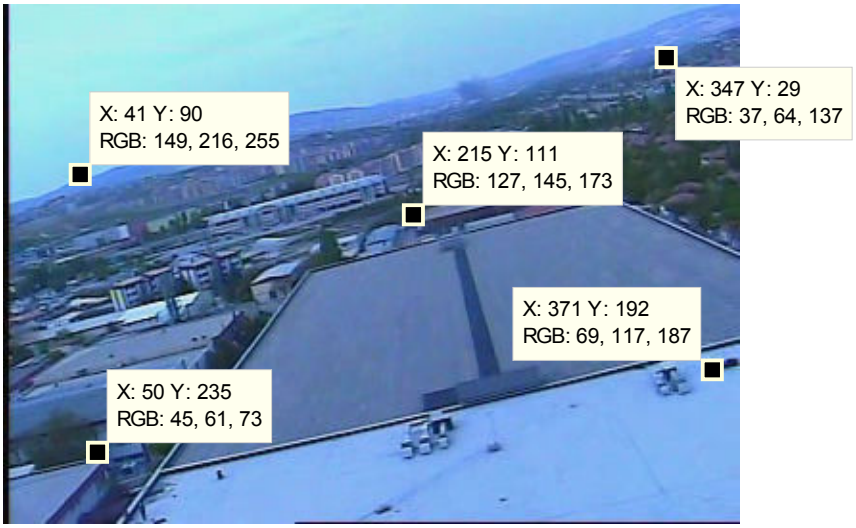
Figure 4: Input frames from zeppelin video for Phase Correlation Algorithm

(a) 15th frame (b) 200th frame (Example-1)

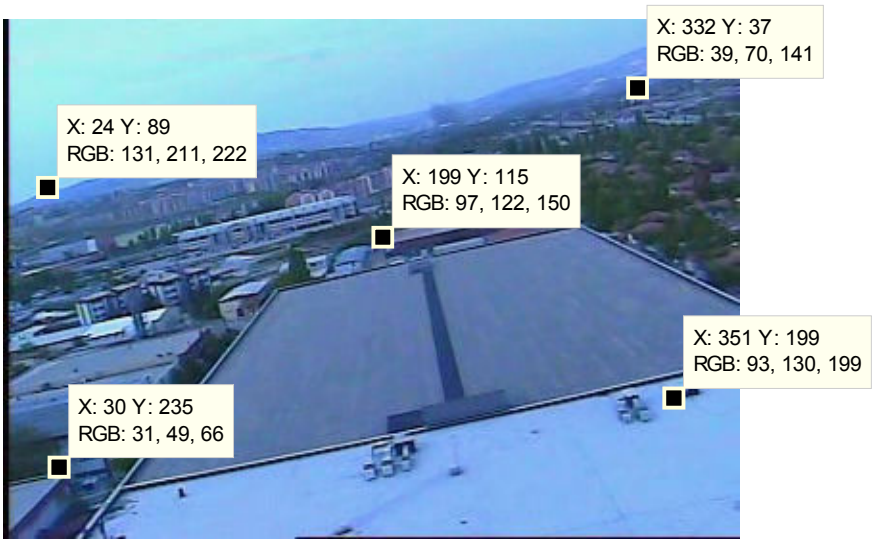
The first real frame example is from zeppelin video. There exists significant displacement between the frames in Figure 4. However, phase correlation algorithm gives zero displacement for these frames in Figure 4

The second real example is again from zeppelin video and frames used in this example have less displacement respect to those of Figure 4, see Figure 5. The phase correlation algorithm gives 1 pixel displacement along x axis.

In order to control the phase correlation result, 5 distinct points are taken on the first frame (15^{th} frame), in Figure 5a. Then corresponding points in the second frame (32^{nd} frame) is found using intensity information of the selected points in Figure 5b. Then, in order to make comparison with phase correlation result, shift for each point is calculated, given in Table 1. The points' displacement results show that there exist rotation around left bottom corner of Figure 5b and there exist more than 10 pixels displacement between the frames 15 and 32. It is seen that the phase correlation method gives very rough result.



(a)



(b)

Figure 5: Input frames from zeppelin video for Phase Correlation Algorithm

(a) 15th frame (b) 32nd frame (Example-2)

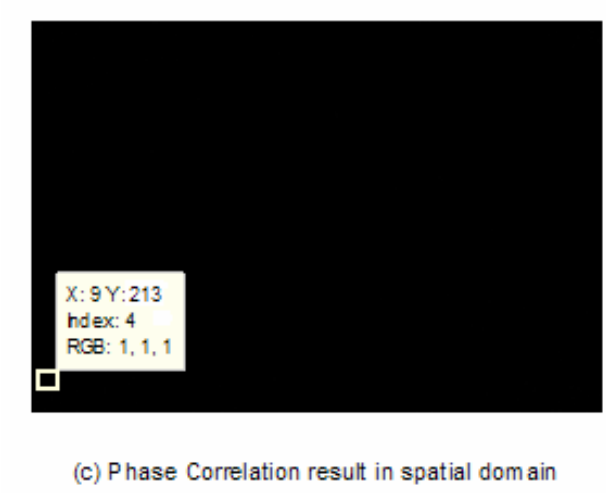
Table 1: Displacement table for Example 2

Point location	Displacement along x: Δx	Displacement along y: Δy
Left bottom	20	0
Right bottom	20	7
Left top	17	1
Right top	15	8
Middle	16	4
Phase Correlation result	1	0

The third example is from "uav" video. Two frames of UAV video are taken and displacement between the frames is calculated by phase correlation algorithms. Then 3 distinct points are taken in the first frame (15^{th} frame), see Figure 6a and corresponding points are found in the second frame (32^{nd} frame), see Figure 6b.. The point displacements and phase correlation result is given in Table 2. There is rotation between the frames around right corner and phase correlation result is not applicable to all points in the frame. Note that, in this video, there are both global and local motions.

(a)

(b)



(c) Phase Correlation result in spatial domain

Figure 6: Input frames from UAV video for Phase Correlation Algorithm

(a) 15th frame (b) 32nd frame (Example-3)

Table 2: Displacement table for Example 3

Point location	Displacement along x: Δx	Displacement along y: Δy
Right bottom	-26	9
Right top	5	2
Left top	12	-21
Phase Correlation result	9	-17

Conclusion

Phase correlation is the only featureless technique implemented in this thesis. In this study phase correlation was planned to be used in prior alignment, i.e. global motion compensation, of the video sequences. However the performance of the algorithm is not satisfactory for the videos which have significant rotation.

Actually, phase correlation is a very fast algorithm, however in the aerial videos where considerable rotation exists, it may not give accurate results. As a result, we decided not to use phase correlation technique in our study.

2.2 Pixel Based Algorithms

In pixel based algorithms, image mosaicing is performed using measurable image quantities of each pixel. The aim is to accurately find the parameters of the transformation matrix between two frames using the information of each pixel. The information extracted may differ according to the algorithm. Pixel intensity values or the optical flow vectors of each pixel can be taken as the measurable quantity.

There are several advantages of utilizing all measurable information in an image. One of them is high precision leveled sub-pixel accuracy which can be exploited to obtain accurate results. Moreover, global motion can be estimated successfully even if outliers exist in the image.

In recent years, instead of utilizing all information in a scene, more informative regions/pixels are exploited in image mosaicing algorithms which have lower computational load and more distinctive information. So, in this study, pixel based algorithms are not implemented instead algorithms which exploit interest points in an image are focused.

CHAPTER 3

FEATURE BASED MOSAICING ALGORITHMS

In feature based mosaicing algorithms more informative regions/pixels are exploited in an image instead of utilizing all information in a scene,. This method brings the advantages of lower computational load and using more distinctive information.

These more informative regions are expressed as the features in an image. There are several criteria for a point to be a feature and in order to obtain an accurate mosaic image, good features should be identified and tracked from frame to frame.

In feature based mosaicing algorithms, features in the reference frame are extracted. Then the correspondent features in the image pair is obtained by either tracking or matching operation. Using the features in the frames, the transformation between the frames is calculated. And the non-reference frame is warped using the transformation matrix and added to the mosaic. Flow chart of

this procedure including the implemented algorithms for each step is given in Figure 7.

So performance of mosaicing system quite depends on selection of appropriate features. This is because tracking/matching performances of the algorithms are highly dependent to the feature properties. So, every feature can not work well in all tracking/matching algorithms. For example the features extracted in the reference frame are matched to the points in the next frame with a window search algorithm using some matching criteria. However if the search window is not attached to a fixed point in the physical world, this makes the feature useless or even harmful to most structure from motion algorithms. Since feature selection is very critical in the way of constructing a good mosaic, three different feature selection algorithms are attempted, see Figure 7.

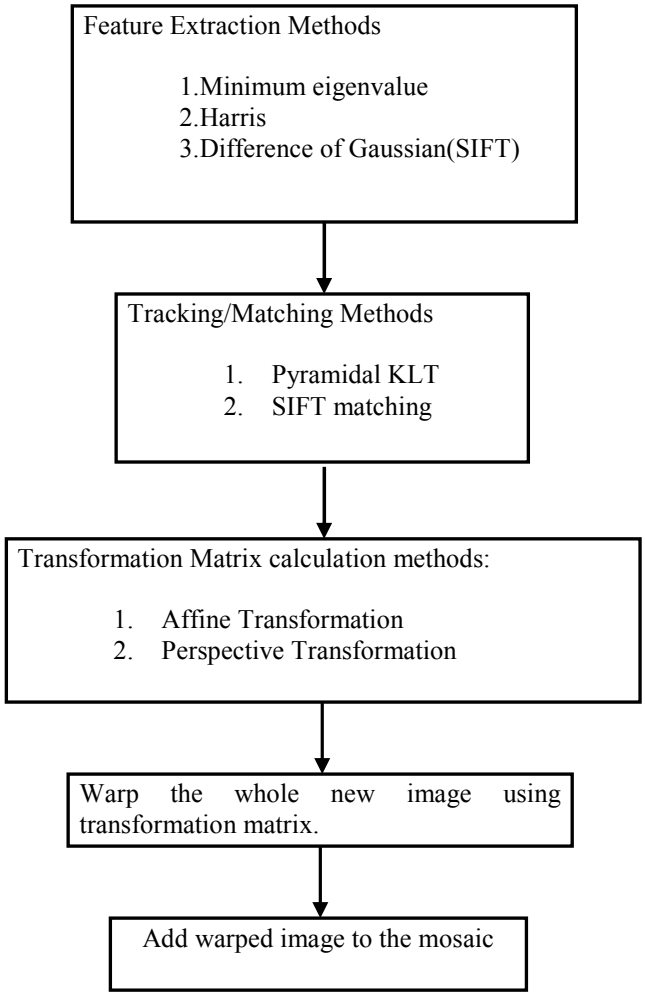


Figure 7: Flow chart of a mosaicing algorithm

Finding the accurate correspondent points of the very first feature points in the new frame is the second step. In tracking/matching part, two algorithms are realized, see Figure 7. The first one is optical flow based KLT algorithm. In this method the correspondent feature points are found by searching the next frame within a window. The point giving the highest correlation is taken as the correspondent feature point. In order to obtain an accurate and robust result, KLT is implemented in pyramidal and iterative form. The second algorithm is SIFT (Scale Invariant Feature Transform) matching. In this method, the features in both of the images are extracted and the closest feature vector in the non reference image is taken as the matching point. Some constraints are used in order to obtain a good matching performance.

After extracting the features in both of the frames, the transformation between the frames is calculated. Transformation is actually a coordinate conversion that maps the image coordinates **x** to the coordinates of reference frame **x'**.

$$\boldsymbol{x} = [x,y]^T \xrightarrow[\text{Matrix}]{\text{Transformation}} \boldsymbol{x'} = [x',y']^T$$

In this study, two kinds of transformation matrix are implemented, as shown in Figure 7. The first matrix utilized is 6 parameter affine transformations where only translation and rotation information is taken into account. The second matrix is the homography matrix in which perspective changes are taken into consideration. The transformation matrixes are calculated by least squares

method. Then the refinement procedure of the transformation matrix is performed. In order to calculate more accurate transformation matrix parameters RANSAC and LMedS algorithms are implemented. In addition to these, transformation matrix found by least square method is refined iteratively.

After finding the transformation between the image pairs, the non-reference frame is warped using these matrix parameters to register the images.

Up to this point transformation matrix which is obtained using only feature points is utilized to warp the whole image. The last step is to update the mosaic image by adding new warped image to the panoramic image. This update process can be performed by adding either the whole new frame or just the new points.

In the next sections of this chapter, implemented mosaicing algorithm steps are explained in a detailed manner.

3.1 Feature Extraction

3.1.1 Feature Extraction Algorithms

In this study, three kinds of feature extraction methods are implemented. The algorithms are listed as follows:

1. Minimum Eigenvalue Method
2. Harris corner detector
3. Difference of Gaussian (SIFT)

3.1.1.1 Minimum Eigenvalue Method

In Minimum Eigenvalue Method, features extraction is performed by the criteria of good traceability. This means that if a matched feature pair can be found reliably, the feature has a good quality.

There exist some problems in feature extraction process. For example, in an image, not all parts of the image contain complete motion information which is known as aperture problem, i.e. motion can be detected only perpendicular to the orientation of the moving contour. Further processing is required to disambiguate true motion direction. To overcome this problem, generally either corners or regions have higher mixture of second order derivatives are selected. Nevertheless, this can not completely solve the problem, e.g. feature

determination window size is undetermined. In Minimum Eigenvalue Method a more principled definition of feature quality is proposed and a good feature is described as the one that can be tracked well, so that the selection criterion is adjusted optimally in construction process [3].

Feature extraction is realized utilizing every pixel in the image. Let I(x,y,k) and I(x,y,k+1) be two gray scaled images and I(**x**) = I(x,y) is grayscale values at location $\mathbf{x} = [x\ y]^T$, where x and y are pixel coordinates of a generic image point **x**.

Consider image points $\mathbf{x_1}$ and $\mathbf{x_2}$ on the first and second images respectively:

$$\mathbf{x_1} = [x\ y]^T \in I(x,y,k)$$

$$\mathbf{x_2} = \mathbf{x_1} + \mathbf{d} = [x+d_x,\ y+d_y]^T \in I(x,y,k+1) \quad (8)$$

where $I(\mathbf{x_1})$ and $I(\mathbf{x_2})$ are the image points and the vector $d = [d_x\ d_y]^T$ is the image displacement at **x**. The goal of this algorithm is to find points in I(x,y,k) that can be tracked well in the second frame I(x,y,k+1). In order to track a point well, the matched point should be found precisely and residual function ε should be minimized.

$$\varepsilon(\bar{d}) = \varepsilon(d_x, d_y) = \sum_{x-w_x}^{x+w_x} \sum_{y-w_y}^{y+w_y} \left(I(x,y,k) - I(x+d_x, y+d_y, k+1)\right)^2 \quad (9)$$

where w_x and w_y are two integers defining the window size. This size determines the area of search window. Because of the aperture problem, notation of

similarity should be defined in 2D neighborhood sense. In order to minimize the residual function ε, gradient of ε(**d**) is equated to zero vector.

$$\left.\frac{\partial\varepsilon(d_x,d_y)}{\partial\bar{d}}\right|_{\bar{d}=\bar{d}_{opt}} = -2\sum_{x+w_x}^{x+w_x}\sum_{x+w_x}^{x+w_y}(I(x,y,k)-I(x+d_x,y+d_y,k+1))\left[\frac{\partial I}{\partial x}\frac{\partial I}{\partial y}\right]=[0 \quad 0] \quad (10)$$

yields to

$$\frac{1}{2}\frac{\partial\varepsilon(\bar{d})}{\partial\bar{d}} \approx \sum_{x-w_x}^{x+w_x}\sum_{y-w_y}^{y+w_y}\left(\nabla I^T\bar{d}-\delta I\right)\nabla I^T \quad (11)$$

where δI is image difference in the search window:

$$\delta I = I(x,y,k)-I(x,y,k+1) \quad (12)$$

Expanding the derivative:

$$\frac{1}{2}\left[\frac{\partial\varepsilon(\bar{d})}{\partial\bar{d}}\right]^T \approx \sum_{x+w_x}^{x+w_x}\sum_{y+w_y}^{y+w_y}\left(\begin{bmatrix}\frac{\partial I^2(x,y)}{\partial^2 x} & \frac{\partial I(x,y)}{\partial x}\frac{\partial I(x,y)}{\partial y}\\ \frac{\partial I(x,y)}{\partial x}\frac{\partial I(x,y)}{\partial y} & \frac{\partial I^2(x,y)}{\partial^2 y}\end{bmatrix}\bar{d}-\begin{bmatrix}\delta I\cdot\frac{\partial I(x,y)}{\partial x}\\ \delta I\cdot\frac{\partial I(x,y)}{\partial y}\end{bmatrix}\right) \quad (13)$$

Denoting :

$$G = \sum_{x-w_x}^{x+w_x}\sum_{y-w_y}^{y+w_y}\left(\begin{bmatrix}\frac{\partial I^2(x,y)}{\partial^2 x} & \frac{\partial I(x,y)}{\partial x}\frac{\partial I(x,y)}{\partial y}\\ \frac{\partial I(x,y)}{\partial x}\frac{\partial I(x,y)}{\partial y} & \frac{\partial I^2(x,y)}{\partial^2 y}\end{bmatrix}\right) \quad (14)$$

$$\bar{b} = \sum_{x-w_x}^{x+w_x} \sum_{y-w_y}^{y+w_y} \begin{pmatrix} \delta I \cdot \dfrac{\partial I(x,y)}{\partial x} \\ \delta I \cdot \dfrac{\partial I(x,y)}{\partial y} \end{pmatrix} \tag{15}$$

Using denotation in Equation (14) and (15), Equation (13) can be rewritten as:

$$\frac{1}{2}\left[\frac{\partial \varepsilon(\bar{d})}{\partial \bar{d}}\right]^T \approx G\bar{d} - \bar{b} \tag{16}$$

Then the optimum displacement vector is:

$$\bar{d}_{opt} = G^{-1}\bar{b} \tag{17}$$

For detailed analysis, refer to [10].

In order to make Equation (16) valid, G should be invertible. From this, we can derive the first feature point criterion to satisfy: The point **x** in the first image should contain gradient information in both x and y directions i.e. none of the gradients is allowed to be zero because even if one of the gradients is 0, this makes the determinant of the matrix 0.

Before giving the details, firstly analyze some critical properties of the gradient matrix, G. This matrix has the following properties:

1. G is a symmetric 2x2 matrix.
2. If eigenvalue decomposition is applied to the gradient G matrix, the eigenvalues (λ_1, λ_2) and eigenvectors $(\vec{e}_1, \vec{e}_2)$ are obtained

respectively. The first eigenvector $\vec{e}_1$, a unit vector, has a direction normal to the gradient edge, while the second eigenvector $\vec{e}_2$ is tangent to the gradient edge. The eigenvalues indicate the underlying certainty of the gradient structure along their associated eigenvector directions.

These gradient features allow a more precise description of the local gradient characteristics. Based on the magnitudes of the eigenvalues, the following inferences can be made based on this argument:

1. If $\lambda_1 \approx 0$ and $\lambda_2 \approx 0$, there are no features of interest at the pixel (x,y).

2. If $\lambda_1 \approx 0$ and λ_2 is some large positive values, then an edge is found.

3. If λ_1 and λ_2 are both large, distinct positive values, then a corner is found.

If Equation (17) gives good measurements and if it can be solved reliably the feature can be tracked from frame to frame. So, the G matrix must be both above the image noise level and well-conditioned. The noise requirement implies that both eigenvalues of the gradient matrix must be large, while the conditioning requirement means that they cannot differ by several orders of magnitude. The eigenvalue magnitudes give idea about the feature profile. Two small eigenvalues mean a roughly constant intensity and a large and a small eigenvalue correspond to a unidirectional texture pattern. On the other hand, two large eigenvalues can represent corners.

In practice, when the smaller eigenvalue is sufficiently large to meet the noise criterion, the G matrix is usually also well conditioned. The pixel is accepted if the two eigenvalues of G are λ_1 and λ_2, are above a threshold.

$$\min(\lambda_1, \lambda_2) > \lambda \tag{18}$$

where λ is a predefined threshold. In fact, the intensity variations in a window are bounded by the maximum allowable pixel value, so that the greater eigenvalue cannot be arbitrarily large.

To determine lower bound of λ, firstly, the eigenvalues for images of a region of approximately uniform brightness is taken with the camera to be used during tracking. Then to obtain an upper bound for λ, a set of various types of features, such as corners and highly textured regions is selected. “In practice, it is found that the two bounds are comfortably separate, and the value of λ, chosen halfway in-between, is not critical [9]”.

In feature extraction algorithm, in Figure 8, the gradient matrix is computed for every pixel in the image. Then the eigenvalues and the eigenvectors of the matrixes are calculated. The selection of the feature points is performed utilizing these values. The minimum values of the eigenvalues are stored and the maximum value of the eigenvalues, λ_{max}, is selected for comparison. Then the points whose minimum eigenvalues are of %5 or %10 percentage of λ_{max} are kept as candidate feature points. After that, local maximum pixel within a

neighborhood of nxn is retained among candidate feature points. Final rejection of feature points is performed with the criteria of minimum pixel distance. This prevents any two feature points to be too much closer to each other and points that are closer more than a threshold value, d, are removed. The selection is performed with respect to the maximum of minimum eigenvalue. Note that, d and n values are the parameters that have to be optimized in this algorithm. In feature extraction it is not necessary to take a very large integration window in computing the G matrix. In fact, a 3x3 (wx=wy=1) window is sufficient for selection, and should be used. Note that for tracking purposes, this window size (3x3) is typically too small. The image points that satisfy all the criteria are typically “good to track" and these pixels are the selected feature points that are fed to the tracker.

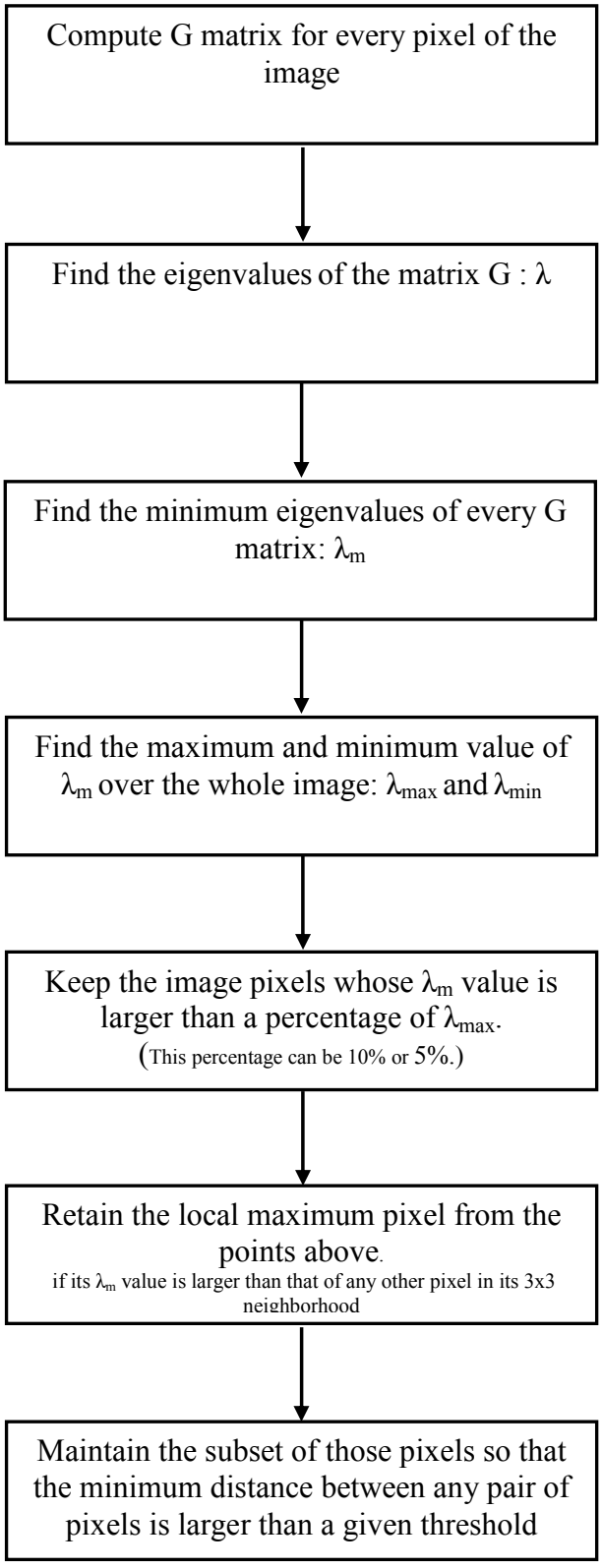


Figure 8: Minimum Eigenvalue Feature Selection Algorithm

3.1.1.2 Harris Corner Detector

Harris Corner Detector [4] is another feature extraction algorithm that finds corner like features. To enable explicit tracking of image features, the image features must be discrete, and not form a continuum like texture, or edge pixels (edgels). For this reason the algorithm is concentrated on the extraction and tracking of feature-points or corners, since they are discrete, reliable and meaningful. However, the lack of connectivity of feature-points is a major limitation in the obtaining of higher level descriptions. So, richer information that is available from edges is needed. Harris corner detector performs this by taking account the differential of the corner score with respect to the direction directly.

Actually, Harris corner detector is an improvement of Moravec's corner detector[18]. It considers a local window in the image and finds the average changes of the image intensity by shifting the window by small amount in various directions. So the algorithm tests each pixel in the image to see if a corner presents. This is done by measuring the similarity of a patch centered on the pixel is to nearby, largely overlapping patches. In other words, Harris relies on the idea that a corner gradient is not well defined at the exact corner location whereas gradient is well defined in the region around the corner. The similarity measurement is the sum of square difference (SSD) between the patches. The lower the difference the higher is the similarity. The points can be grouped in three regarding SSD results: flat region, edge and corner regions. If the SSD gives a small change, the centered pixel is considered to be in uniform region. If

the nearby patches in the direction orthogonal to the patch is quite different and tangent to the patch is similar, then the point can be taken as edge. If the feature has patch variation in all the directions, and no patches are similar, this indicates the presence of a corner point. Note that the response is anisotropic because only a discrete set of shifts at every 45 degrees is considered.

In order to improve the Moravec's algorithm, Harris corner detector takes the gradient of the image into account. The gradient G matrix in Equation (14) is obtained in the same way mentioned in section 3.1.1.1. The point is selected as a feature using a response function R. Harris defines a corner/edge response function, R, to measure the quality of the corner. The response function is calculated using the eigenvalues of the gradient matrix:

$$R = \lambda_1\lambda_2 - \kappa(\lambda_1 + \lambda_2)^2 \qquad (19)$$

where the value of κ is the parameter which has to be determined empirically, and in the literature values in 0.04 - 0.15 range have been reported as feasible.

In order to obtain a rotational invariant result, Equation (19) has to be a function of only eigenvalues, λ_1 and λ_2. It is computationally expensive to make the eigenvalue decomposition of the G matrix. For easier computation, trace, *Tr(G)*, and determinant, *Det(G)*, of the G matrix is utilized where:

$$Tr(G) = \lambda_1 + \lambda_2$$

$$Det(G) = \lambda_1 \lambda_2$$

$$R = Det(G) - \kappa Tr^2(G) \qquad (20)$$

Using the information in R, both corner/edge classification and corner/edge quality determination can be performed. The point in the image is categorized according to the result of R function. If for a given point, R is positive, it is considered as the corner region, if negative as the edge regions, and if small as flat region, (Figure 9). In other words, for a feature to be in corner region, both eigenvalues should be large so is response function. On the other hand, if one of the eigenvalues is small (close to zero) being the other larger, the feature falls into the edge region. Finally, if both of the eigenvalues are small (close or equal to zero), the feature is considered in flat region.

Figure 9 is the graph of the response function in eigenvalue space. Pixels belong to edge region have large λ_1 small λ_2 (or vice versa), ideally one of them being zero. But in reality one of the eigenvalues is merely small compared to other one due to noise, pixellation and intensity quantization. In the same way, a pixel is considered as edge if it has both negative response and local minima in either x or y directions (thin edge). A corner is the point where there is two large eigenvalues, i.e. a pixel is selected as corner if it has 8-way local maximum. Similarly, flat region pixels have small eigenvalues. However, what is the reference of being small and large eigenvalues?

The threshold determines whether the pixel belongs to flat or edge/corner regions is found experimentally. In determination of this threshold edge hysteresis is observed by applying low and high thresholds. This procedure is used for enhancement of the continuity of edges [4].

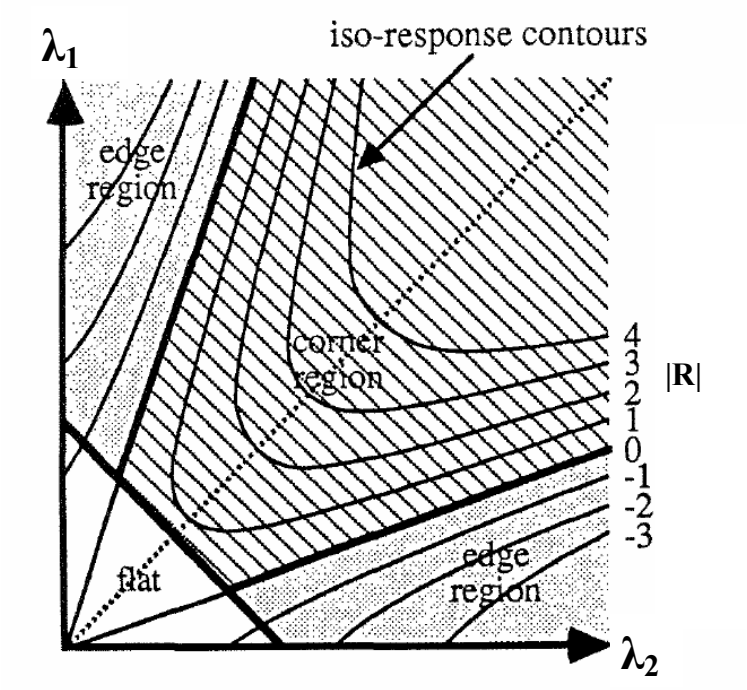


Figure 9: Amplitude of the Response Function, |R|.

The block schema of the Harris corner detector is given in Figure 10. In summary, in Harris corner detector algorithm, G matrix is computed for every pixel in the image and utilizing the eigenvalues or determinant and trace of the matrix a response function is obtained. The corner decision is performed if the

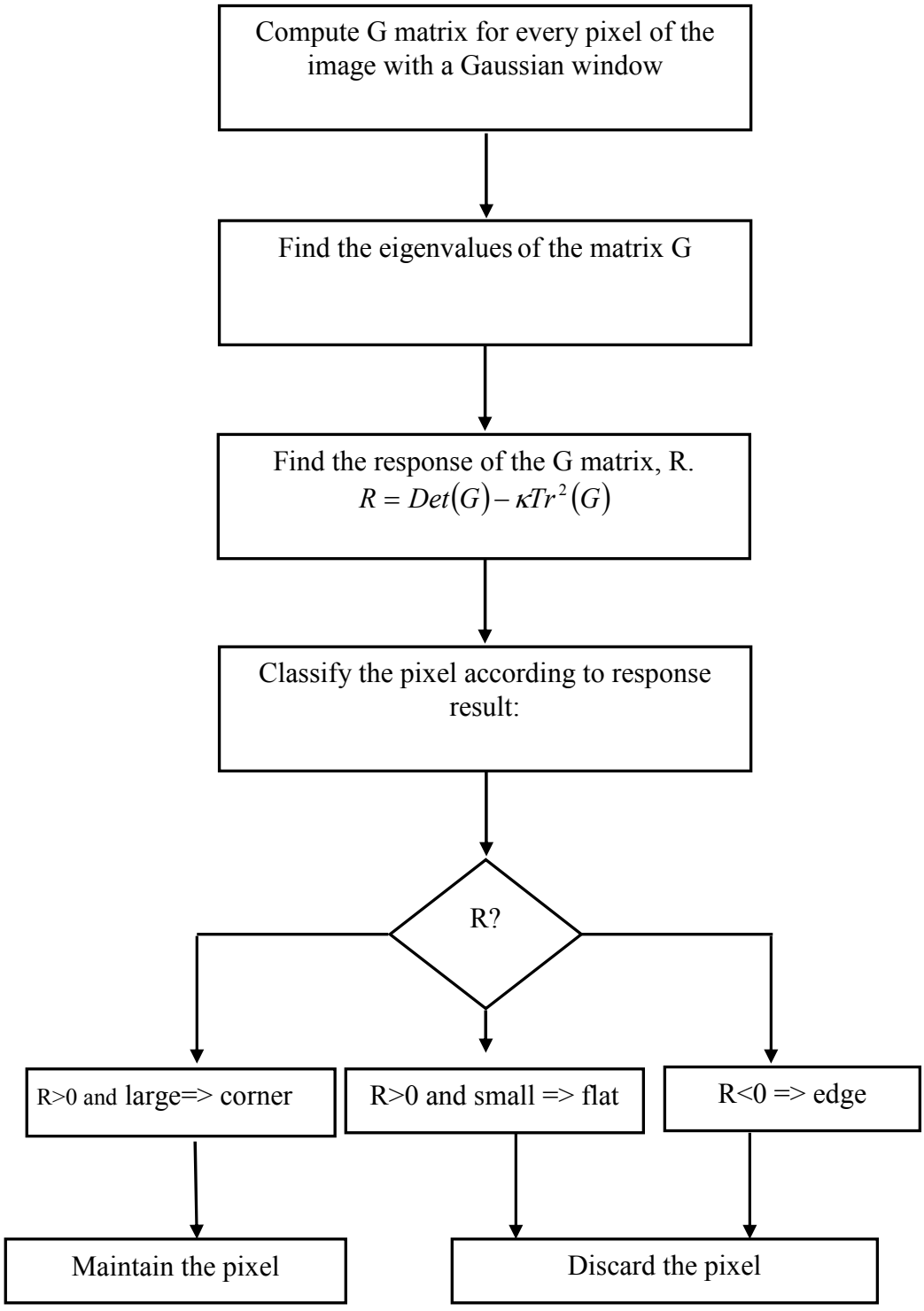


Figure 10: Harris Feature Detection Algorithm

response, R, has a positive and high value. Note that G matrix is obtained using a window and the size of the window is critical. If the window is small, variation is smaller and sensitive to noise, so less reliable. On the other hand, a large window can make it smooth. As in the minimum eigenvalue method 3x3 window is preferred in feature selection.

3.1.1.3 SIFT Keypoint Extraction by Difference of Gaussian Method

Keypoint Extraction by Difference of Gaussian in Scale Invariant Feature Transform (SIFT) is an algorithm for extraction of distinctive features in an image. The power of this algorithm is that the features obtained are fully/partially invariant to scale, rotation, viewpoint and illumination changes. The algorithm realizes this by transforming the image data into scale-invariant coordinates relative to local features.

This algorithm utilizes the Difference of Gaussian kernel for scale invariant feature detection. Note, this kernel is invariant to scale and orientation. Then the algorithm finds the local maximum of difference of Gaussians in space and scale. In this way, the keypoint candidates are detected using a cascade filtering. In the first step, the original image is Gaussian blurred progressively with σ ranging from 1 to 2 in the way of identifying locations and scales that can be repeatably assigned under different views of the same object. The scale space of an image is defined as a function, L:

$$L(x, y, \sigma) = Gs(x, y, \sigma) * I(x, y) \tag{21}$$

where *Gs* is the Gaussian Function. Then the difference-of-Gaussian function, separated by a constant multiple of k, is convolved with the image:

$$D(x, y, \sigma) = (Gs(x, y, k\sigma) - Gs(x, y, \sigma)) * I(x, y) \tag{22}$$

Then maxima and minima of the difference of Gaussian images are found by neighborhood comparison method. Given a pixel (x,y) in an image, $D(x,y,\sigma)$ is compared with its 26 neighbors at the current and adjacent scales. 26 pixels correspond to 8 neighbors in the current scale and 9 neighbors each in the two adjacent scales obtained by multiplying by different σ values. The point is chosen if it is minimum or maximum of the neighbor pixels.

After finding the keypoint candidates, the second step is to perform accurate keypoint location. In this step low contrast and poorly localized points are eliminated. This is realized by fitting a 3 dimensional quadratic function, Taylor expansion up to the quadratic terms of $D(x,y,\sigma)$, to the local sample points. In this way, the interpolated location of the maximum is found in a stable way. The function value at the maximum, $D(\hat{x})$, is utilized for rejecting unstable extremas with low contrast.

$$D(\hat{x}) = D + \frac{1}{2}\frac{\partial D}{\partial x}^{T} \hat{x} \tag{23}$$

The rejection is performed for the points with a value of |D(x)| less than 0.03.

In addition to elimination of low contrast values, the points which are located poorly along an edge have to be rejected. This is performed by using of Hessian matrix (square matrix of second-order partial derivatives) of D. Utilizing the trace and determinant information of the Hessian the elimination is performed.

Then as the third step, orientation assignment is performed for each keypoint. Histograms of gradient directions are computed in a 16x16 window using bilinear interpolation. In order to assign orientation in a scale invariant manner, the scale of the keypoint is used to select the Gaussian smoothed image, L. The gradient magnitude, m(x,y) and the orientation θ(x,y) are computed as:

$$m(x,y) = \sqrt{\left(\left(L(x+1,y) - L(x-1,y)\right)^2 + \left(L(x,y+1) - L(x,y-1)\right)^2\right)} \quad (24)$$

$$\theta(x,y) = \tan^{-1}\left(\frac{L(x,y+1) - L(x,y-1)}{L(x+1,y) - L(x-1,y)}\right) \quad (25)$$

The orientation histogram is constructed using the gradient orientations of sample points within a region around the keypoint. It has 36 bins and covers the 360 degree range of orientations. Each sample is weighted by its gradient magnitude and by a Gaussian-weighted circular window with an σ that is 1.5 times that of the scale of the keypoint and added to the histogram [6].

The forth step is computing local image descriptors which is a representation in-a 128 dimensional vector. Firstly, the gradient magnitude and orientation at each image sample point in a region around the keypoint location is computed. Then a

Gaussian window is used to weight these values. These samples are then accumulated into orientation histograms summarizing the contents over 4x4 sub regions, with the length of each arrow corresponding to the sum of the gradient magnitudes near that direction within the region 4x4 descriptors computed from a 16x16 sample array[6]. Utilizing all these orientation histogram values, descriptor is constructed, which is a 128 dimensional feature vector. Finally, the illumination change effect on the descriptor vector is reduced.

The power of the keypoints extracted by SIFT algorithm is that they enable the correct match from a large database of keypoints with its property of high dimensional vector representing the image gradients within a local region. Besides, the keypoints are invariant to rotation and scale and robust to illumination change, noise and affine distortion. The matching part of the SIFT algorithm is explained in feature matching section, 3.3.4.

3.2 Feature Tracking

Feature tracking is one of the most fundamental operations in computer vision. Feature tracking methods can be categorized into two:

1. Dense Techniques
2. Sparse Techniques

Dense techniques extract features from each frame and then attempt to establish correspondences between both set of features. This approach requires that the same feature is detected reliably and consistently between consecutive frames. The disadvantage of this algorithm is that correspondence errors tend to be very large. SIFT algorithm which uses DoG (Difference of Gaussians) is an application of dense based algorithms.

On the other hand, sparse techniques extract features only from the reference frame. The location of the features in subsequent frames is found by performing a global search inside a suitable sized window. The point which correlates best with the texture around the feature in the reference frame is taken as the correspondent. The disadvantage of this technique is that features tend to drift. They also do not cope well when the texture in the subsequent frame has been rotated, zoomed or skewed with respect to the texture in the first frame. Pyramidal KLT algorithm is an application of sparse- algorithms.

3.2.1 Pyramidal Iterative Lucas and Kanade

Pyramidal Lucas and Kanade is a motion estimation method in computer vision. It is used to find the correspondent feature points in the image pair. Pyramidal KLT is based on optical flow concept which assumes the constancy of intensity/color between two frames. Considering constant luminance assumption in 2D:

$$I(x, y, t) = I(x + dx, y + dy, t + dt) \tag{26}$$

Assuming small δ values, apply Taylor series expansion:

$$I(x + dx, y + dy, t + dt) = I(x, y, t) + \frac{\partial I}{\partial x} dx + \frac{\partial I}{\partial y} dy + dt + H.O.T. \tag{27}$$

Ignoring higher order terms (H.O.T.) and taking derivative with respect to t:

$$\frac{\partial I}{\partial x} V_x + \frac{\partial I}{\partial y} V_y + \frac{\partial I}{\partial t} = 0 \Rightarrow \nabla I \bullet \vec{V} = -I_t \tag{28}$$

where $V_{x,y}$ indicates the velocity along x,y directions, respectively. Equation (28) has two unknowns, and needs another constraint to be solved, known as aperture problem. In this point, Lucas and Kanade approach suggests a solution to the problem which assumes motion vectors remain unchanged over a particular block of pixels of size mxm. Using this information, an over determined system is obtained and solved.

In pyramidal iterative KLT, the solution is obtained iteratively. In order to find the correspondent point in the next window, it is aimed to minimize Equation (16) in a robust and accurate manner. For accurate result a pyramidal approach in which local sub-pixel accuracy can be attached to tracking and small integration window is preferred. This is performed not to smooth out the details. This approach becomes especially critical at occluding areas in the images where two patches potentially move with very different velocities. On the other hand, in order to obtain a robust solution, the result should not be sensitive to noise and should be independent of some effects such as light changes, scale of motion etc. In addition to these, the integration window size should be large enough for the algorithm to handle large motion. So, in choosing the integration window, there is a tradeoff between local accuracy and robustness. Pyramidal KLT approach is a solution to this problem. Additionally, iterative implementation of the algorithm makes the local accuracy higher.

The pyramidal representation of the image is:

$$\begin{aligned} I_L(x,y) &= \frac{1}{4} I_{L-1}(2x,2y) + \\ &\frac{1}{8}\left(I_{L-1}(2x-1,2y) + I_{L-1}(2x+1,2y) + I_{L-1}(2x,2y-1) + I_{L-1}(2x,2y+1)\right) + \\ &\frac{1}{16}\left(I_{L-1}(2x-1,2y-1) + I_{L-1}(2x+1,2y+1) + I_{L-1}(2x-1,2y+1) + I_{L-1}(2x+1,2y+1)\right) \end{aligned} \tag{29}$$

where I_L is the L^{th} level image. The original image I_0 has a level of "zero" and a size of (n_x x n_y). The pyramid representation is constructed recursively. From the

last step, I_0 is computed from I_1, I_1 from I_2 … I_{L-1} from I_L (Equation (29)) and so on. Similarly, n_x and n_y^{L-1} are the width and height of the image, I_{L-1}.

In pyramidal KLT, the correspondence point of a given feature $\mathbf{x_1}$ in the reference image is found by finding the optimum displacement vector **d** in Equation (17). For typical image sizes, it makes no sense to go above a level 4. The central motivation behind pyramidal representation is to be able to handle large pixel motions. Therefore the pyramid height should also be picked appropriately according to the maximum expected optical flow in the image. The overall pyramidal tracking algorithm proceeds as follows: First, the optical flow is computed at the deepest pyramid level I_{Lm}. Then, the result of that computation is propagated to the upper level I_{Lm-1} in a form of an initial guess for the pixel displacement (at level I_{Lm-1}). Given that initial guess, the refined optical flow is computed at level I_{Lm-1}, and the result is propagated to level I_{Lm-2} and so on up to the level 0 (the original image). Observe that the window of integration is of constant size $(2w_x + 1)$ x $(2w_y + 1)$ for all level values [10].

The pseudo-code of the algorithm is given in the Figure 11.

In this study, the matching points are found in sub pixel accuracy level. This is because both displacement vector and center points are not guaranteed to be integers. To calculate the sub pixel intensity values, bilinear interpolation is used.

$$I_L(x,y) = (1-\alpha_x)(1-\alpha_y)I_L(x_0,y_0) + \alpha_x(1-\alpha_y)I_L(x_0+1,y_0) + (1-\alpha_x)\alpha_y I_L(x_0,y_0+1) + \alpha_x\alpha_y I_L(x_0+1,y_0+1) \quad (30)$$

The final part of tracking is the decision section. What is the criterion for elimination of a feature? It is obvious that if the feature is outside the boundary, it is eliminated. However if the point is close to boundary so that a portion of the integration window falls outside the image, the feature is kept and procedure is performed with the window inside the image boundary.

If the patch difference is too much, than the point is assumed not to be tracked well and then discarded. Note that, there has to be made a decision of which difference correspondence to a large value. This threshold value is very critical in especially long sequence of frames.

For more detailed information of Figure 11 and whole equation, see [10].

***Goal**: Let $\mathbf{x}_1$ be a point on image I(k). Find its corresponding location $\mathbf{x}_2$ on image I(k+1)*

Build pyramid representations of I(k) and I(k+1) : $\{I(k)^L\}_{L=0,\ldots L_m}$ and $\{I(k+1)^L\}_{L=0,\ldots L_m}$

Initialization of pyramid guess: $g_{L_m} = [g_{x L_m} \quad g_{x L_m}]^T = [0 \quad 0]^T$

for L = L_m down to 0 with step of -1

Location of point x_1 on image I_L : $u_L = [p_x \quad p_y]^T = u / 2^L$

Derivative of I_L with respect to x : $I_x(x,y) = \dfrac{I_L(x+1,y) - I_L(x-1,y)}{2}$

Derivative of I_L with respect to y : $I_y(x,y) = \dfrac{I_L(x,y+1) - I_L(x,y-1)}{2}$

Spatial gradient matrix:

$$G = \sum_{x=p_x-w_x}^{p_x+w_x} \sum_{y=p_y-w_y}^{p_y+w_y} \begin{bmatrix} I_x^2(x,y) & I_x(x,y)I_y(x,y) \\ I_x(x,y)I_y(x,y) & I_y^2(x,y) \end{bmatrix}$$

Initialization of iterative L-K: $\bar{x}_2^{\,0} = [0 \quad 0]^T$

for k=1 to K with step of 1 (or until $\|\bar{\eta}^k\|$ < accuracy threshold)

Image difference:

$$\delta I_k(x,y) = I_L(x,y) - J_L(x + g_{x L} + x_2^{\,k-1}, y + g_{y L} + y_2^{\,k-1})$$

Image mismatch vector:

$$\bar{b}_k = \sum_{x=p_x-w_x}^{p_x+w_x} \sum_{y=p_y-w_y}^{p_y+w_y} \begin{bmatrix} \delta I_k(x,y) I_x(x,y) \\ \delta I_k(x,y) I_y(x,y) \end{bmatrix}$$

Optical flow (Lucas-Kanade): $\bar{\eta}^k = G^{-1}\bar{b}_k$

Guess for next iteration: $\bar{v}^k = \bar{v}^{k-1} + \bar{\eta}^k$

end of for-loop on k

Final optical flow at level L: $d_L = x_2^{\,K}$

Guess for next level L-1 : $g_{L-1} = [g_{x L-1} \quad g_{y L-1}]^T = 2(g^L + d^L)$

end of for-loop on L

Final optical flow vector: $d = g^0 + d^0$

Location of point on I(k+1): $x_2 = x_1 + d$

***Solution:** The corresponding point is at location $\mathbf{x}_2$ on image I(k+1)*

Figure 11: Pseudo-code of the Pyramidal KLT algorithm

3.2.2 SIFT Matching Algorithm

Scale invariant feature matching algorithm [6], is a method for extracting distinctive invariant features from images which allows performing reliable matching between different views of an object or scene. In section 3.1.1.3, feature extraction part of the algorithm is mentioned. The SIFT features are highly distinctive that allows a single feature to be correctly match with high probability against a large database of features. In this part, this matching approach of the SIFT algorithm is exploited.

In general, SIFT algorithm compares the feature with every feature in the new image and finds candidate matching features based on Euclidean distance of their feature vectors by nearest-neighbor algorithms. However finding the correct match with high probability is difficult especially in cluttered images. In order to recover the incorrect matches, the correct matches are filtered by identifying subsets of keypoints that agree on the object as well as its location, scale and orientation in the new image. So, the incorrect match probability is decreased since the probability that several features will agree on these parameters by chance is much lower than the probability that any individual feature match will be in error. The determination of these consistent clusters can be performed rapidly by using an efficient hash table implementation of the generalized Hough transform. After this filter, a new verification is performed for features that agree on an object and its pose. First, a least-squared estimate is made for an affine approximation to the object pose. Any other image features consistent with this

pose are identified, and outliers are discarded. Finally, a detailed computation is made of the probability that a particular set of features indicates the presence of an object, given the accuracy of fit and number of probable false matches. Object matches that pass all these tests can be identified as correct with high confidence.

3.3 Transformation Matrix and Mosaic Image Construction

Up to this part features in the reference image are extracted and matching points in the image pair are found. In this part, a model is fitted to these feature points by two different ways:

1. Affine Transformation
2. Homography Transformation

The model selection is very critical since it is applied to the whole image and the non-reference image is warped using this transformation model. It is important to mention that as the parameter of the model increases the accuracy does not increase. This is due to the fact that, solving the equation becomes difficult and as the time complexity increase the overall mosaicing performance decreases. On the other hand, small number of parameters can not model the motion and changes. So, optimum parameter number has to be selected.

3.3.1 Transformation Matrix Model

3.3.1.1 Six -6- Parameter Affine Transformation

Affine transformation is a linear transformation. This model contains translation, rotation and scale. Physically, affine model preserves collinearity between points, i.e. three points which lie on a line continue to be collinear after the transformation.

Affine Transformation: $\mathbf{x}' = \mathbf{A}\mathbf{x} + \mathbf{b}$ (31)

In the aerial videos, transformation matrix should handle camera pan and tilt. However affine transformation can not express this kind of changes. So the world cannot be modeled accurately.

As mentioned before, transformation matrix is calculated from feature points in the very first and new frame. So, all these features' movements are represented only by 6 parameters for affine transformation. Since the motion of the features in the frame is more complicated than linear transformation (rotation, scaling and shear) and translation, affine model is inadequate for aerial videos.

3.3.1.2 Eight -8- Parameter Homography Transformation

Homography transformation is an 8 parameter matrix transformation. It performs transformation to from image coordinates **x** to reference image coordinates **x**'..

$$\begin{bmatrix} x' & y' & 1 \end{bmatrix}^T = \overline{H} \begin{bmatrix} x & y & 1 \end{bmatrix}^T \tag{32}$$

In Equation (32), **H** is a 3x3 matrix. Let **H** matrix be:

$$H = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \tag{33}$$

The last parameter of the H matrix is equated to 1 in order to make the matrix normalized. Besides homography matrix accept planar surface assumption.

Then image coordinates are calculated as:

$$x' = \frac{\left(a_{11}x + a_{12}y + a_{13}\right)}{\left(a_{31}x + a_{32}y + 1\right)}$$

$$y' = \frac{\left(a_{21}x + a_{22}y + a_{23}\right)}{\left(a_{31}x + a_{32}y + 1\right)} \tag{34}$$

As mentioned, the transformation matrix is calculated using all the feature points in both frames. So, the displacement of the whole image should be represented using number of transformation matrix parameters and 8-parameter projective model gives the enough accuracy to account for all the possible camera motions and perspective changes.

So, for the aerial videos, 8-parameter homography matrix is adequate to compensate for the camera motions and mosaic image is obtained using 8-parameter homography matrix. The parameters are calculated with minimizing SSD error.

3.3.2 Transformation Matrix calculation Algorithms

3.3.2.1 Least Squares

Least squares algorithm calculates the transformation matrix by solving an over determined function using standard least squares method. In this method, the coordinates of the feature points in the image pair is given, and an optimum transformation matrix between the coordinates of the features is calculated by standard least squares method.

3.3.2.2 RANSAC

RANSAC is the algorithm to estimate the mathematical model of a set of observed data. Observed data contains both inliers and outliers, where inliers correspond to a set of data that can be described by some set of parameters whereas outliers can not be described by a model. So, for an accurate model fitting, these outliers have to be eliminated.

In general, analysis of interest points have two problems: First problem is to find, the best match between the data and one of the available models (the classification problem); Second problem is to compute the best values for the free parameters of the selected model (the parameter estimation problem). In practice, these two problems are dependent and a solution to the parameter estimation problem is often required to solve the classification problem.

In the way of fitting a model to a set of parameters, classical methods utilize all the parameters. However such methods have no internal mechanism to detect/ reject errors. To recover this, several heuristics have been proposed. However, it is seen that even a single gross error in the set of good data cause this technique to fail. On the other hand, RANSAC is capable of fitting inliers to a model even if the data set contains large set of gross errors.

The RANSAC paradigm is more formally stated as follows:

Given a model that requires a minimum of n data points to instantiate its free parameters, and a set of data points P such that the number of points in P is greater than n. The model is instantiated by randomly selecting a subset SI of n data points from P. The instantiated model M1 is used to determine the subset SI* of points in P that are within some error tolerance of Ml. The set SI* is called the consensus set of S1. If g(SI*) is greater than some threshold t, which is a function of the estimate of the number of gross errors in P, use SI* to compute (possibly using least squares) a new model MI*. If g(SI*) is less than t, randomly select a new subset S2 and repeat the above process. After a predetermined number of trials, if no consensus set with t or more members has been found, either solve the model with the largest consensus set found, or terminate in failure.

In the RANSAC algorithm, number of iteration, q, is a very critical parameter. Let z be the probability of any selected data which is in the error tolerance and $b=z^n$. Then, E(q) can be written:

$$E(q) = b + 2*(1-b)*b + 3*(1-b)^2*b + \ldots\ldots + i*(1-b)^{i-1}*b + \ldots\ldots \qquad (35)$$

. Using the geometric series properties:

$$E(q) = \frac{1}{b} = z^{-n} \qquad (36)$$

In order to stay in the safe side, generally it is preferred to exceed E(k) trials by one or two standard deviations which is defined as:

$$SD(q) = \left[E\left(q^2\right) - E(q)^2\right]^{1/2} = \left[\left(1 - z^n\right)^{1/2}\right] * \left(1/z^n\right) \quad (37)$$

On the other hand, if a predetermined probability, c, is wanted to be ensured, k is chosen as:

$$k = [\log(1-c)]/[\log(1-b)] \quad (38)$$

In RANSAC, the decision of terminating the program is given if number of inliers is above a predetermined threshold, t. This threshold must be chosen large enough to satisfy two purposes:

1. The correct model has been found for the data set,

2. Sufficient number of mutually consistent points has been found to satisfy the needs of the final smoothing procedure (which computes improved estimates for the model parameters) [7].

The flowchart of the implemented RANSAC algorithm is given
Figure 25.

3.3.2.3 LMedS

Least Median of Squares Method [5] is another technique to estimate a model to a set of observed data. In general, data derived from images is non-homogenous, so can not be described by a unique parametric model. The observed data in an image usually contains small scale noise as well as occasional large-scale measurement errors. LMedS is an approach to estimate an accurate and robust

model which can handle up to 50% of outliers i.e. breakpoint of the algorithm is 0.5. Actually, there are several measures of robustness. One of the most common one is the breakdown points which is described as the minimum fraction of outlying data that can cause an estimate to diverge arbitrarily far from the true estimate.

The idea behind LMedS is that median of data is less sensitive to outliers compared to the algorithms which use mean-like values. LMedS method utilizes the median of the error function for calculations:

$$\hat{e} = \arg\min_{e} \left(\underset{x_i \in X}{median} \left(r^2{}_{i,a} \right) \right) \tag{39}$$

where $\hat{e}$ is the error estimation of SSD function.

In the algorithm, a model is obtained using minimum necessary number of data points. Then the model is applied to every pixel to estimate the parameter and an error function is obtained:

$$r^2{}_{i,a} = \left(x - \hat{x} \right)^2 \tag{40}$$

In the implemented LMedS algorithm, the feature points are bucketized and a model is fitted to those randomly selected points. Then for every pixel, the SSD is calculated for every pixel in the image. The median of these values are taken and compared to a threshold found experimentally. After classifying the data, the model is tested according to the number of inliers and desired probability, P_g:

$$P_g = 1 - \left(1 - p^k\right)^S \tag{41}$$

where S is the number of subsets which is critical and has to be large enough to have high probability of including at least one subset containing all "good" data points is the minimum fraction of good points, p is the minimum fraction of all good points, k is the minimum number of points required to fit the model.

The iteration goes on until the desired probability and corresponding S value is obtained.

.

CHAPTER 4

RESULTS AND COMPARISON OF MOSAICING ALGORITHMS

In previous chapters, the steps of mosaic image construction algorithms are explained. In this chapter, the performance and comparison of the implemented algorithms are given.

In this study, three different feature extractor methods are implemented and two of them are used in the implemented algorithms. The first feature extractor method is “Harris Corner Detector” and the second one is SIFT Keypoint Detector which uses Difference of Gaussian method. In this study, “Minimum Eigenvalue Method” is implemented as the third method. However since the performance of this method is lower than Harris Corner Detector, only Harris features are examined. Besides, affine transformation model is found insufficient for aerial videos in which perspective changes occur. So, in the resultant

algorithms only the homography model which handles perspective changes is used.

In this study, 4 algorithms are implemented as given in Table 3 and each algorithm is tested for 4 different videos. The first two are real videos, zeppelin and uav, and mosaic results for these videos are presented. The other two videos are synthetically generated and more detailed performance analysis is done for these “wall” and “amasra” videos. The original full frame images are also given in Figure 12 for the synthetically generated videos which allow pixel wise differencing of real image and mosaic results. Besides SNR values for the last two videos are calculated which gives an idea about visual performance of the algorithms. Time complexity of each algorithm is also given.

Table 3: Steps of Implemented Algorithms

		Algorithms			
		MA1	MA2	MA3	MA4
Algorithm Steps	Feature Extraction	Harris Corner Detector	Harris Corner Detector	Harris Corner Detector	Difference of Gaussian(DoG)
	Feature Tracking/Matching	Pyramidal KLT Tracker	Pyramidal KLT Tracker	Pyramidal KLT Tracker	SIFT matching
	Homography Calculation	Least Squares	RANSAC	LMedS	Least Squares

(a) (b)

(c) (d)

Figure 12: Original full frame images of (a)"wall" (c) "amasra" videos. Transformed Original Images with respect to first frames of (b)"wall" (d)"amasra" videos. Pixel intensity differences are calculated from (b) and (d) images.

The tracking and homography matrix accuracies are measured by synthetically generated images. For this analysis, a frame of “zeppelin” video is captured in Figure 13 and it is warped according to a known homography matrix. Then the frame is warped with respect to the known homography matrix. Then features of the original frame are extracted and tracked. In order to obtained the tracking error, the feature points are transformed with the known homography matrix and difference between the tracking coordinates are calculated as error. The homography matrix accuracy performance is determined by calculating the homography between the tracked and original frames. The results of these procedures are given in related algorithm section below.

In this study, there are 4 algorithms implemented and two performance criteria for the algorithms are used:

1. SNR values
2. Time complexity

Additionally, the parameters which affect the performance of each algorithm are analyzed and algorithms are compared.

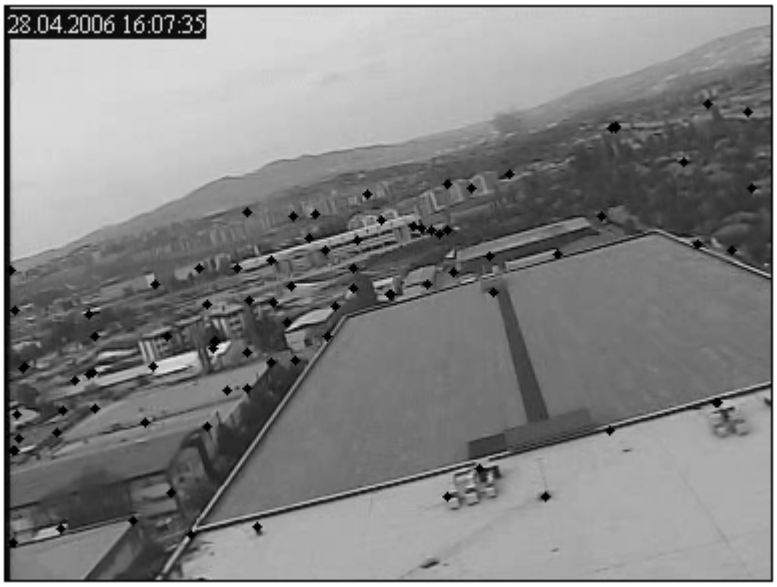


Figure 13: Original Image

4.1 Mosaicing Algorithm 1

Mosaicing algorithm 1 (MA1) includes the following step in mosaic construction:

1. Feature extraction is performed using Minimum Harris
2. Feature tracking is done with Pyramidal KLT Method
3. Homography Transformation Matrix with LS is used

The flowchart of MA1 algorithm is given in

Figure 14. In the algorithm, the very first frame in the video is selected as the reference frame. Then feature properties and mosaic image is initialized in the

"Make initialization" part. The reference image is placed in the middle of the mosaic image which is 3 times larger than the reference frame for showing new frames symmetrically. Features are extracted utilizing "Harris Corner Detector Method". In this study, features extracted with Minimum Eigenvalue Method are analyzed and the results of the Harris is found more stable and tests show that Harris features give more accurate tracking results. For this reason Harris Detector is selected as feature extractor.

In Harris feature extraction process, 2x2 gradient covariance matrix G in Equation.(14), over 3×3 neighborhood is calculated for each pixel,. Then, as mentioned in 3.1.1.2, $[\det(G) - \kappa * \text{trace}(G)^2]$ is calculated for each pixel. As a result of this operation, candidate corners in the image are selected from the points which have local maxima of these difference values. Here, κ is parameter which has a range of (0.04-0.15). And the studies show that, lower the parameter, better the results. So, all the results utilizing Harris Corner Detector are obtained with the κ value of 0.04.

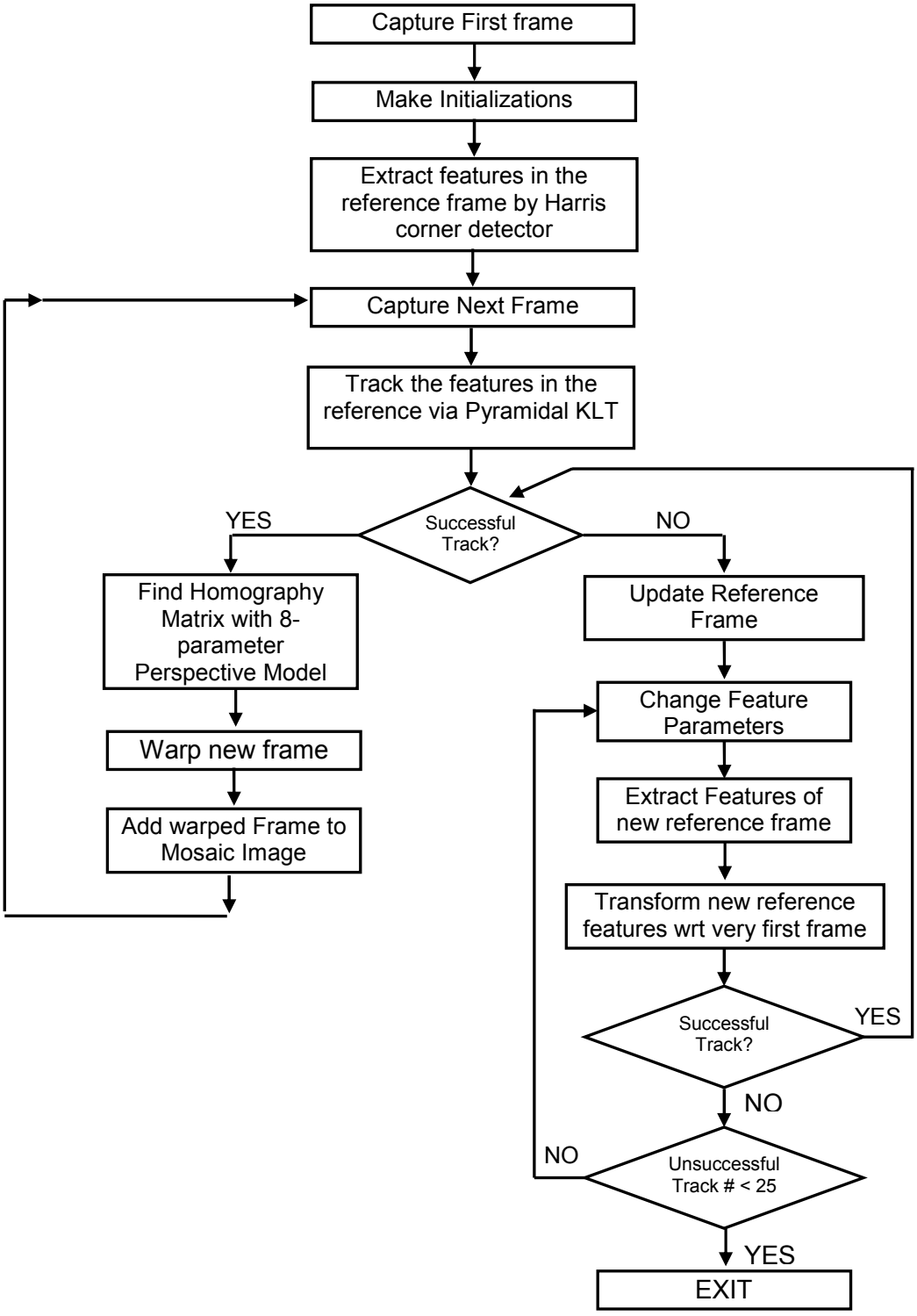


Figure 14: Flowchart of Mosaic Algorithm 1

The next step is rejecting the corners with respect to their properties. In feature elimination from candidate features process, there are two parameters which have to be optimized with respect to the video characteristics. These are:

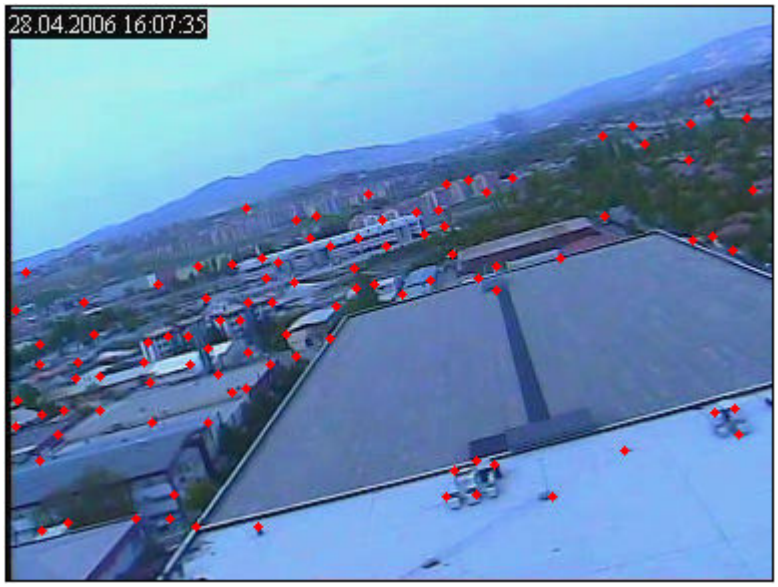
1. Multiplier factor of maximum eigenvalue: min-eig
2. Maximum distance between two features: min-dist

In rejection candidate features, the feature that has the max eigenvalue is determined and the points having the eigenvalues less than [min-eig •max(eig_image(x,y))] are eliminated. The second criterion is the pixel wise distance between the features coordinates. The points are eliminated in such a way that the Euclidean distance between the features coordinates are larger than min-dist.. So, less strong features around the strongest features are removed. Figure 16 shows the extracted features in reference frames of all 4 videos utilized in this study. The parameters of this step are very critical in mosaicing performance. In order to obtain accurate mosaic image, the features extracted have to be robust and instable points have to be eliminated as much as possible. On the other hand, features have to be distributed uniformly in order to give the perspective to the mosaic correctly. The parameters used are given in Table4.

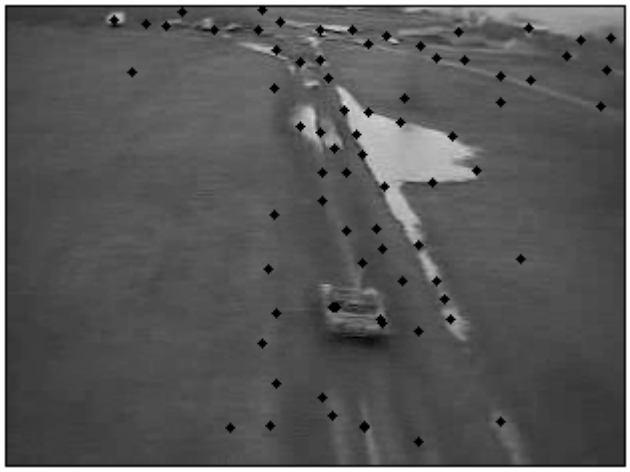
Table 4: Minimum distance and eigenvalue parameters for the videos

		VIDEO			
		Zeppelin	Uav	Wall	Amasra
Parameter	Min-dist	10	10	10	10
	Min-eig	0.1	0.01	0.1	0.1

Tests show that, if the corner-like features present in the frames of video, min-dist :10 - min-eig: 0.1 gives enough number of features. On the other hand, if the corner-like features are not distributed uniformly in the image, these values have to be decreased in a suitable manner. As an example, in the uav video, the objects in the image are mostly located in the center portion of the frame. So, in order to get correct perspective, more points have to be selected which are uniformly distributed over the image. So the min-eig value is decreased to tenth of the selected number. Besides, the size of the image has to be considered in especially selection of the min-dist parameter. The number of features extracted in the reference frames of each video is given in Figure 16.



(a)



(b)

Figure 15: Features of Harris Method in real videos (a) zeppelin (b) uav

(a)

(b)

Figure 16: Features of Harris Method in synthetic videos (a) wall (b) amasra

After features are extracted in the reference frame, the matching points of these features are found in the next frame by iterative version of Lucas-Kanade optical flow in pyramids. The coordinates of the feature points on the current video frame given their coordinates on the previous frame are calculated with sub-pixel accuracy. The parameters in the tracking algorithm are:

1. The tracking window size: win-size
2. Pyramid level
3. Iteration Number and pixel difference Size of the search window of each pyramid level

Win-size determines the size of the search window of each pyramidal level in the way of finding the corresponding point in the next image. So if the frame difference between the consecutive video frames is large, the size has been selected larger. For the pyramidal level 3 levels is enough for lower resolution video such as aerial videos, Third parameter determines the stopping criterion of the iteration. The KLT is stopped if (number of iterations) > 20 or (pixel difference) < 0.03. The tracking parameters used in the mosaic construction is given in Table 5.

Table 5: Parameters of KLT of the videos

		VIDEO			
		Zeppelin	Uav	Wall	Amasra
Parameter	Win-size	3	15	5	3
	Pyr-level	3	3	3	3
	ItNum- dif	20 - 3	20 - 3	20 - 3	20 - 3

The tracking window size of the videos is different. This is because, the videos have different characteristics. Zeppelin video has small displacement between the consecutive frames whereas uav has larger displacement. In order to find the corresponding feature in the following frame for video1, small window size has to be chosen. On the other hand, if video has larger displacement, a larger window size should be chosen in order to track features in the video sequence. The KLT tracker's performance decreases significantly as the displacement increases between the image pairs. In order to measure the tracking performance of KLT, a synthetic image pair is generated as mentioned before. The first frame in Figure 13 is filtered with the given homography matrix and the second image in Figure 17 is generated. The features of the images are extracted with Harris Detector and difference between the feature coordinates is calculated. Figure 18 shows the tracking error differences along-x and along-y directions. As seen from the Figure 18, tracking error is on the order of 0.5 pixels which is a well result.

Note that the tracking error is the same for first three algorithms since all utilize KLT tracker

Figure 17: Cropped Image of Figure 13

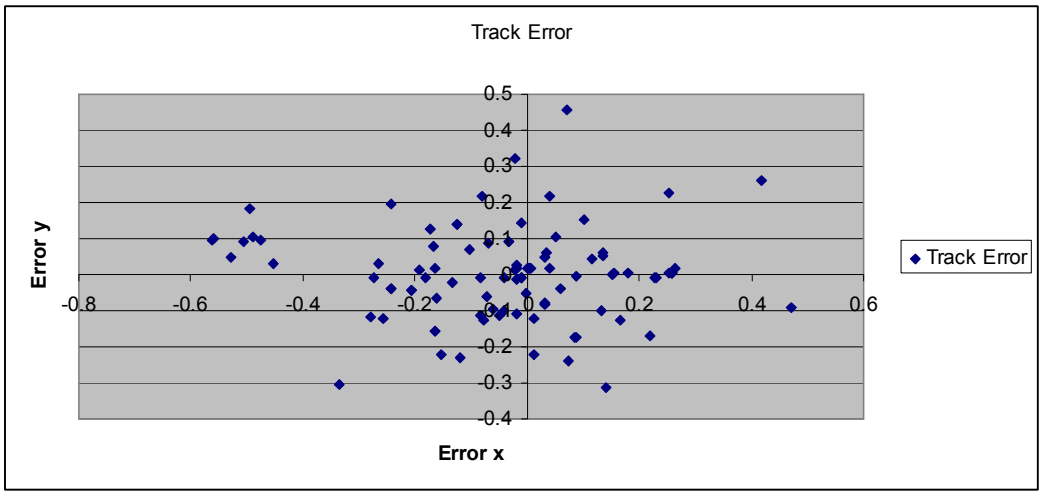


Figure 18: Track error of KLT in x and y directions

After extracting the feature by Harris Corner Detector and finding the corresponding features in the next frame by KLT tracker, the transformation between the feature points of the reference frame and next frame are calculated. In this study, perspective distortion is assumed between frames so homography matrix calculation is performed by means of least squares method following an SSD minimization.

As new frames come, the difference between the reference frame and the new captured frame increases which means overlapping area between the frames decreases. So, in tracking operation, number of matching features decreases due to the movement of the camera. On the other hand, in order to calculate the correct perspective between the frames, features have to be made distributed all over the image. So, in order to provide uniform distribution of the features, the reference frame is updated. The update process is performed if the number of features is below a threshold. This is a very critical parameter in the mosaic construction process. The threshold value (feature threshold) selected for each video is given in Table 6. As seen from Table 6, the threshold number is very close to the number of features in the reference frame as it should be in order to calculate the correct perspective.. This means that reference frame update is performed even small number of features are not tracked in the new frame. Frequent reference frame update process prevents to localization the features in an image.

Table 6: Number of features and thresholds

	VIDEO			
	Zeppelin	Uav	Wall	Amasra
Feature Number	97	105	170	175
Feature Threshold	93	95	160	170

Reference frame is updated to previous frame and new features of it are extracted. In this point a second control mechanism works. If the number of extracted features in the new reference frame is below feature threshold, perspective can be lost in the next frames. So, feature extraction process in the new reference frame is repeated until number of features reaches to an above level. The number of features is increased by decreasing min-eig and min-dist values, respectively. When the number of tracked points in the new frame is above threshold, the same procedure is applied and homograph matrix is calculated utilizing all these feature point pairs by least squares. There can be incorrect feature pairs in the images. However in this algorithm, this issue is not considered. The homography matrix accuracy of Least Squares Method is given in Figure 19.

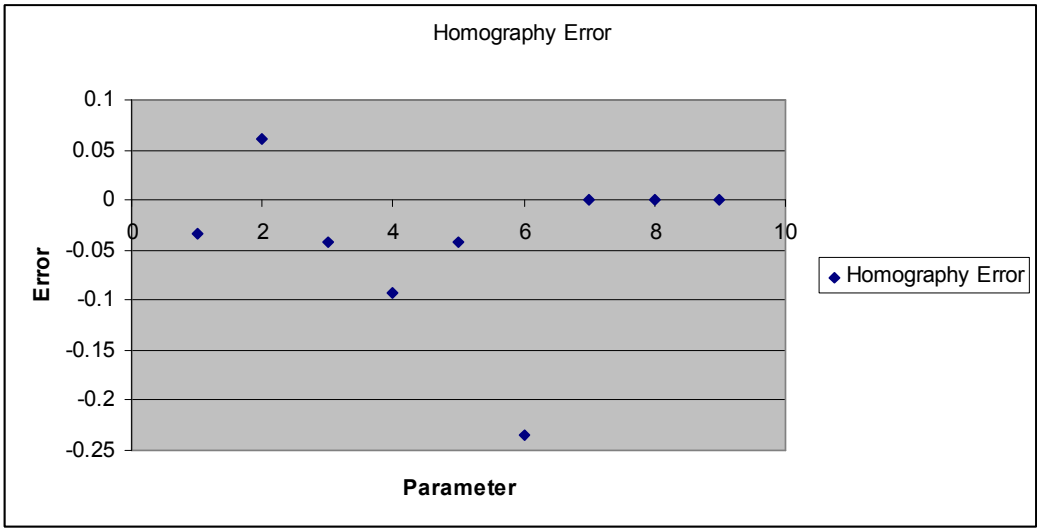


Figure 19: Homography Parameter Error of Least Squares

The error is obtained from synthetic image pairs. After calculating the features in the original and cropped images, the homography matrix parameters are calculated. Figure 19 shows the difference of each 9 parameters of homography matrix.

After homography matrix is calculated between the reference and new frame, the last step is to transform the new coming image with respect to the reference frame and adding to the mosaic image. The mosaic results for real and synthetic videos are given in Figure 20 and Figure 21 respectively.

The mosaic images in Figure 21 are constructed by just adding of only the new parts to the previous mosaic. This is for providing a better visualization and recognizing the performance of the mosaicing process easily.

The SNR performance of MA1 for the synthetically generated videos is given in Table 7.

Table 7: SNR for the synthetic videos

	Video	
	Amasra	Wall
SNR value	27.5606	41.42

The SNR calculation is performed with respect to Equation (42).

$$SNR = 20\log\frac{255}{\sqrt{SSD}} \tag{42}$$

The Sum of Squared Difference (SSD) value is calculated as Equation (43):

$$SSD = \sum_{\forall x,y}\left(I_{ref}(x,y) - I_{mosaic}(x,y)\right)^2 \tag{43}$$

(a)

(b)

Figure 20: Mosaic Results of Mosaicing Algorithm 1 for real videos

(a) zeppelin (b) uav

(a)

(b)

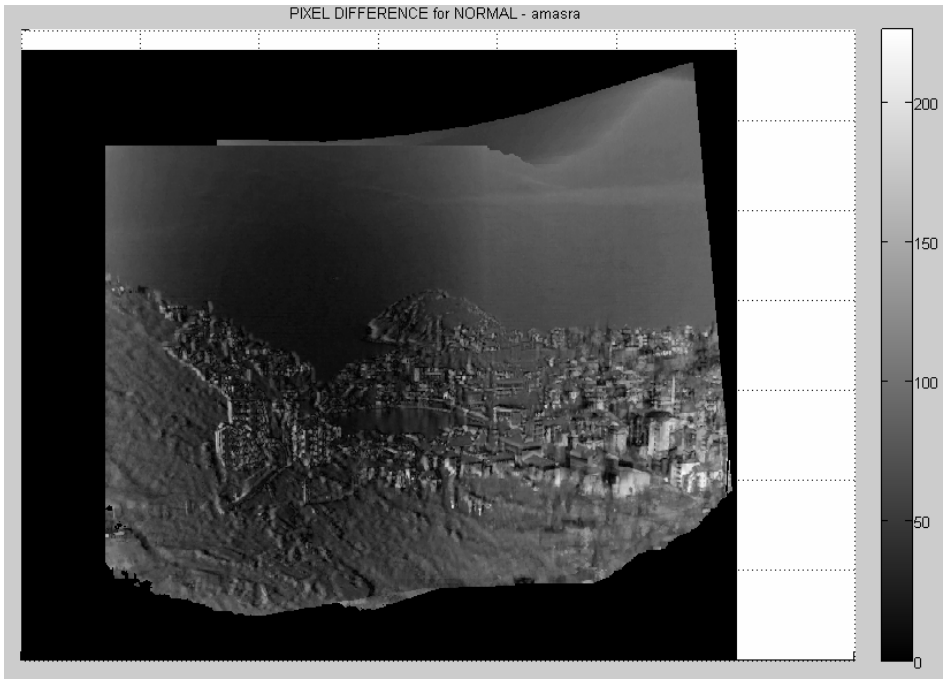
Figure 21: Mosaic Results of Mosaicing Algorithm 1 for synthetic videos

(a) wall (b) amasra

The pixel-wise difference of the mosaic images in Figure 21 and actual images in Figure 12 for the synthetically generated videos are given in Figure 22. In this figure, the intensity difference is calculated for every pixel in the image (if there is a correspondence in actual image) and the difference values are plotted for every pixel. Using the information of SNR and results in Figure 22, it is recognized that the wall mosaic has a worst result. This is because of the video characteristic. In “wall” video, the features are not distributed uniformly. Most of the features in the wall image are concentrated on the painting on the right and increasing the number of features does not handle the problem since it increases the number of features in this painting. Besides, there is much more motion difference in “wall” video than “amasra” video. This means that the difference between the consecutive frames is large which decreases the tracking performance and cause misalignments. In addition to these factors, empty areas in the wall image results to lose the perspective.

Considering the SNR values, wall mosaic has higher value which generally results better visual performance. However, from pixel differences, it is seen that wall mosaic has worse results. This is because of the existence of the constant intensity wall background in the image. In this video, most of the errors mosaicing are not taken into account since wall has almost the same intensity which results a higher SNR rate.

(a)



(b)

Figure 22: Pixel difference of mosaic image and original image in (a)wall (b)amasra videos

4.2 Mosaicing Algorithm 2

The Mosaicing Algorithm 2 (MA2) is composed of the following steps:

1. Feature extraction via Harris Corner Detector
2. Feature tracking with Pyramidal KLT Method
3. Homography Transformation Matrix by RANSAC

MA2 algorithm is very similar to MA1 except for the third step in which transformation matrix calculation is performed. The flowchart of MA2 is given in Figure 24. The feature extraction and feature tracking steps are carried out as the same as of MA1, so the extracted features points are the same as Figure 16. Besides, the parameters for the first two steps are the same for all 4 videos as MA1. The difference between the first two algorithms appears in the calculation of Homography transformation between the feature points of the images. In MA2, it is assumed that some of the feature pairs are incorrect, and these incorrect feature pairs are eliminated by Random Sample Consensus (RANSAC) algorithm.

Implementation of RANSAC is given in
Figure 25 . There are 5 parameters in RANSAC algorithm:

1. Bucket size: n
2. Euclidean difference(distance) threshold: euc-th

3. Inlier threshold
4. Assurance
5. Iteration Number

RANSAC requires the same number of measurements as the number of unknown parameter of the equation. In this situation, there are 8 unknown parameters in homography matrix which means that 4 feature points are enough for RANSAC. The critical point here is the selection of these 4 feature pairs. These pairs have to be selected randomly and they have to be uniformly distributed. For this reason, the image is divided into nxn pairs. In this study, n is selected as 4 since 4 pairs are needed for RANSAC operation. Some tests are performed with more than 4x4 buckets and no improvements are recorded.

After dividing the image, 4 points are selected randomly from different buckets and homography matrix is calculated from these 4 feature-pairs. Then, this homography matrix is applied to other feature points, and corresponding feature location is found in the next frame, Than the Euclidean difference between the actual tracked pair coordinate and homography result is calculated Using this difference, inlier/outlier determination is performed. If the Euclidean difference is below the Euclidean difference threshold, the point is taken as inlier, otherwise considered as outlier. The Euclidean difference threshold parameter is determined by examining the feature characteristics of the video . In determination of this threshold, the Euclidean differences between the feature points are plotted and it is observed that the difference is about 0.7 and more differences might cause

incorrect result so it is selected as 0.7. Then the homography matrix obtained from the first 4 feature pairs is applied to all of the feature points. As a result all features are grouped either as inlier or outlier. Then, utilizing the number of inliers, inlier probability is calculated. This probability value has to be determined by considering the characteristics of the features. According to tracking performance of the points, a rough threshold is determined. The inlier probability of 0.8 is used for all the videos.. After determination of inlier probability, the Iteration Number update is performed utilizing this probability. In this point, the desired assurance of the result is taken into account. As the assurance increases, the iteration number increases which raise the time spent. The assurance is selected as 0.9 and maximum Iteration number is selected as 10000. It is observed that, maximum iteration number increases the mosaic image quality. After updating the Iteration number, another random 4pairs are selected and iteration end until iteration number decreases to 1. After this point, resultant homography matrix is calculated from best feature pairs which give the maximum inlier probability. This method eliminates outliers and calculates homography matrix from the inlier points by least squares which gives a more accurate homography result. The error of homography matrix obtained from synthetically generated images is given in Figure 23.

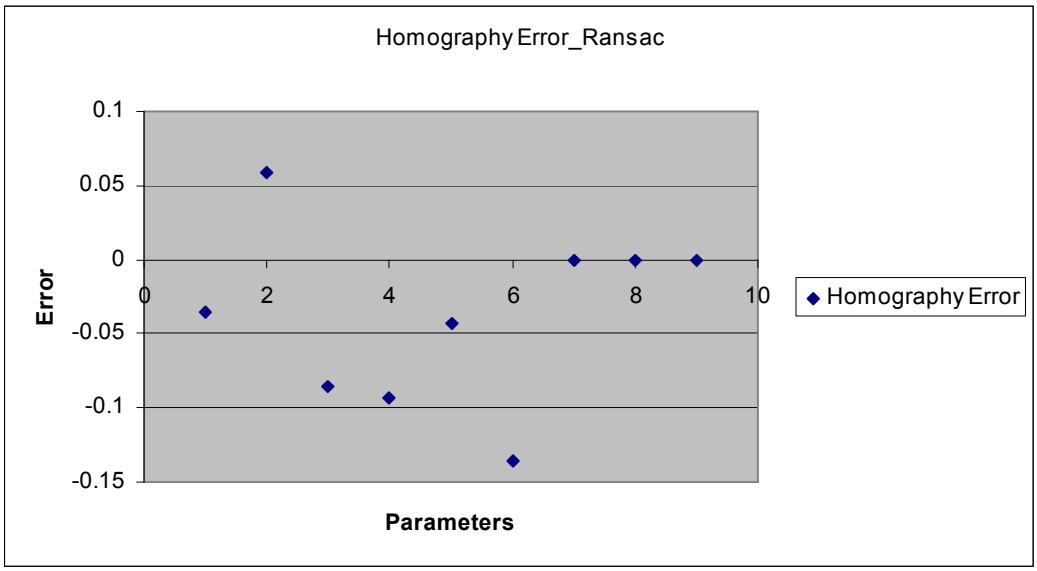


Figure 23: Homography Error for RANSAC

After finding the homography matrix parameters, new frame is warp with respect to this matrix and added to the mosaic image. The mosaic results of Mosaicing Algorithm 2 for real and synthetically generated videos are given in Figure 26 and Figure 27., respectively.

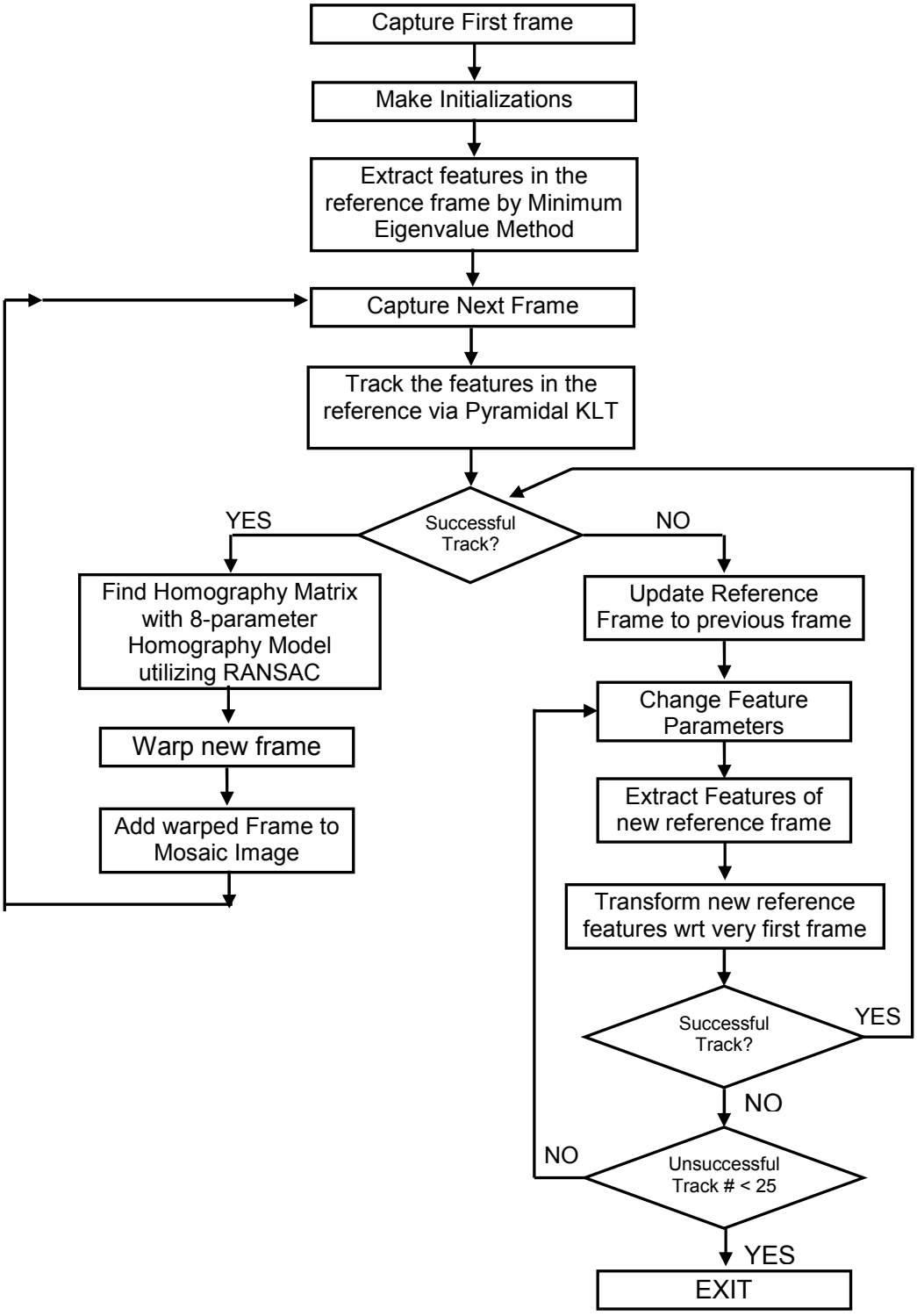


Figure 24: Flowchart of Mosaic Algorithm 2

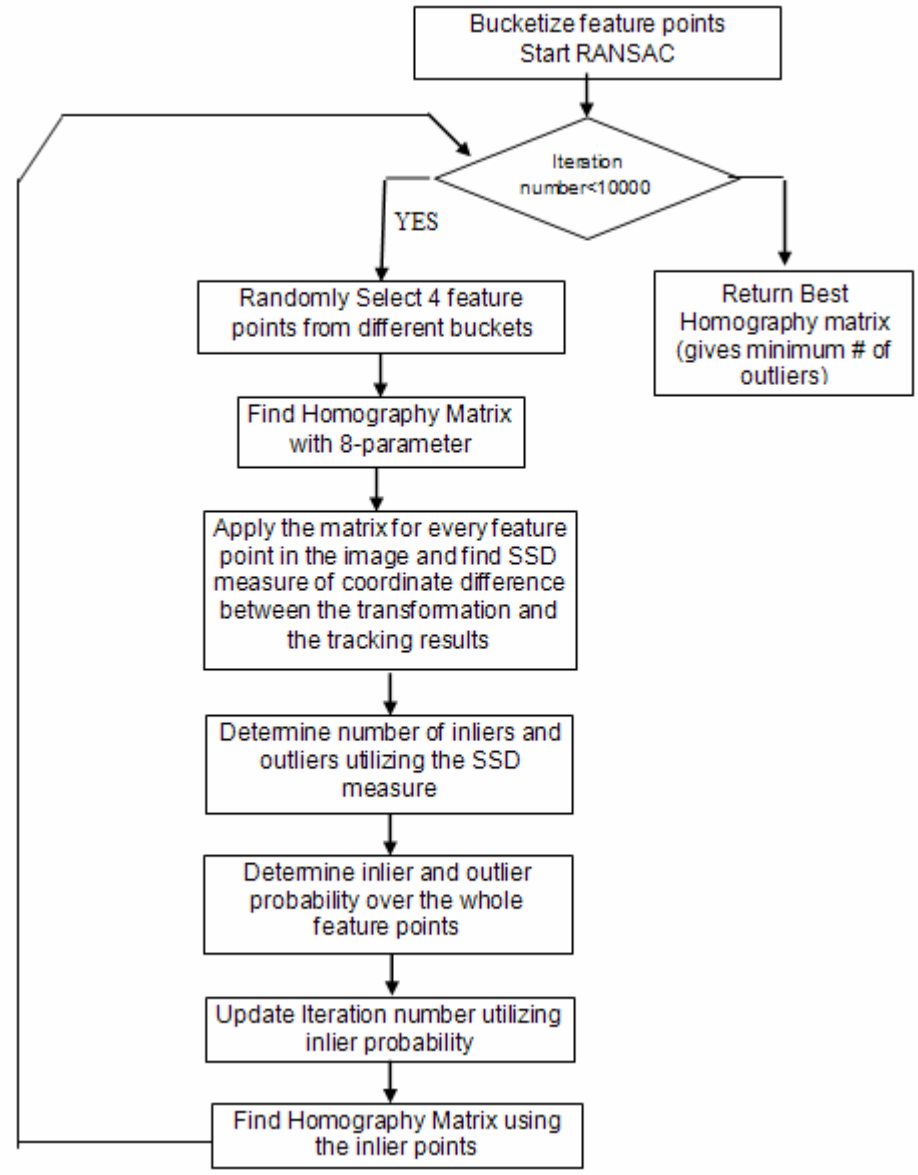


Figure 25: RANSAC Implementation

(a)

(b)

Figure 26: Mosaic Results of Mosaic Algorithm 2 for synthetic videos

(a) zeppelin (b) uav

(a)

(b)

Figure 27: Mosaic Results of Mosaic Algorithm 2 for synthetic videos

(a) wall (b) amasra videos

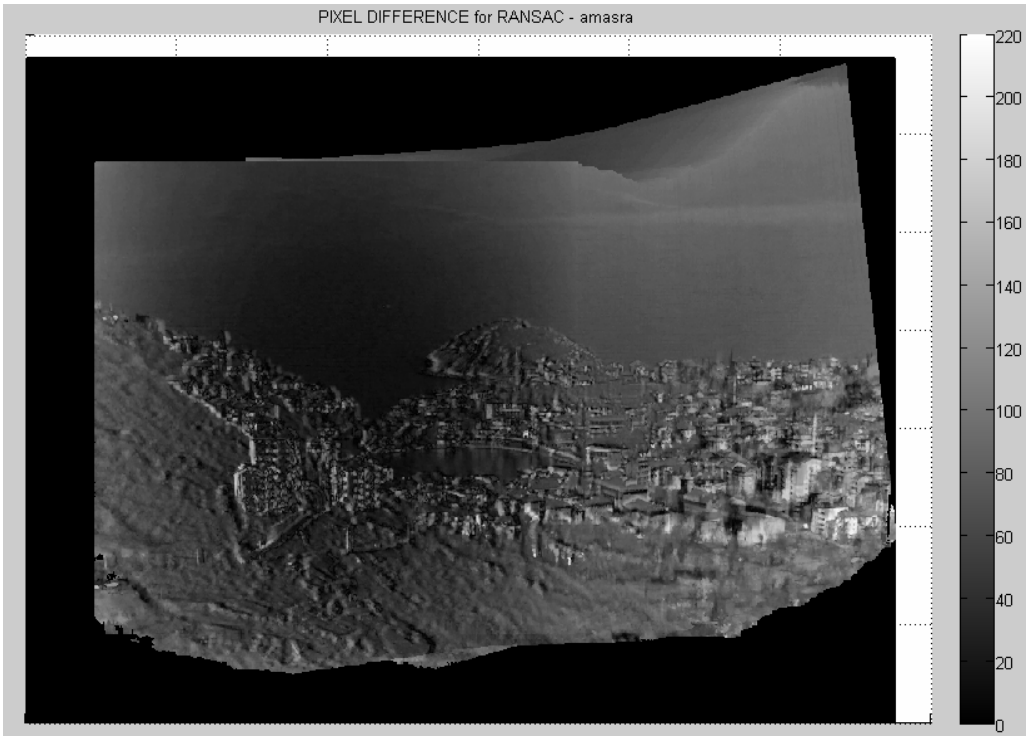
The SNR results of MA2 for the synthetically generated videos are given in Table 8.

Table 8: SNR values for MA3

	Video	
	Amasra	Wall
SNR value	29.52	41.69

The pixel-wise intensity difference between the actual images in Figure 12 and mosaic images in Figure 27 is given in Figure 28.

(a)



(b)

Figure 28: Pixel-wise frame difference for (a) wall (b) amasra mosaic

4.3 Mosaicing Algorithm 3

The Mosaicing Algorithm 3 (MA3) is composed of the following steps:

1. Feature extraction Harris Corner Detector
2. Feature tracking with Pyramidal KLT Method
3. Homography Transformation Matrix with LMedS

The flowchart of MA3 algorithm is given in Figure 30. In MA3, the features are extracted by Harris Corner Detector and the corresponding points in the new frame are found by tracking these features in the next frame by KLT tracker method. These steps are the same as MA1 and MA2 and the parameters used for these two steps are the same for MA3.

After features are extracted for the image pairs, transformation matrix is calculated with Least Median of Squares Method (LMedS). The algorithm is very similar to RANSAC except for best feature pair decision process. Flowchart of the LMedS algorithm is given in Figure 31. There are two parameters of the LMedS algorithm:

1. Bucket size: nxn
2. Iteration Number

In LMedS method, image is divided into nxn regions. N is selected as 4 as in RANSAC. So the image is divided into 16 regions and all features belonging to these portions are assigned to each bucket. Then 4 feature pairs are randomly selected from different buckets and homography matrix is calculated from these

pairs. In order to determine the quality of the selected features, this matrix is applied to every feature in the image and Euclidean differences between the actual tracked points and homography result are stored. After this point, the median of the Euclidean differences is extracted and stored. This procedure is repeated for Iteration times and the feature pairs which gives the smallest median is taken as the reference feature points. Here the Iteration number is critical since it determines the quality of the mosaic result. The Iteration number is selected from the total number of feature point information. Actually iteration number determines the number of 4pairs selected process before decision of best pair. Of course, the best result is obtained if all 4 features pairs are tried. However this increases the time taken for each new frame and slows down the algorithms. Here Iteration number is selected as 2000 which is proper for the feature number if it is around 100.

Then homography matrix is calculated from determined reference feature pairs. So, here the homography matrix is calculated from 4 feature pairs. The critical point of LMedS is, it can reject if number of outliers are less than 50% of the feature points. So examining of tracing performances of the features have to be observed first and this algorithm should be considered if more than 50% of the feature points stay stable. The accuracy of homography matrix obtained from synthetically generated images is given in Figure 29.

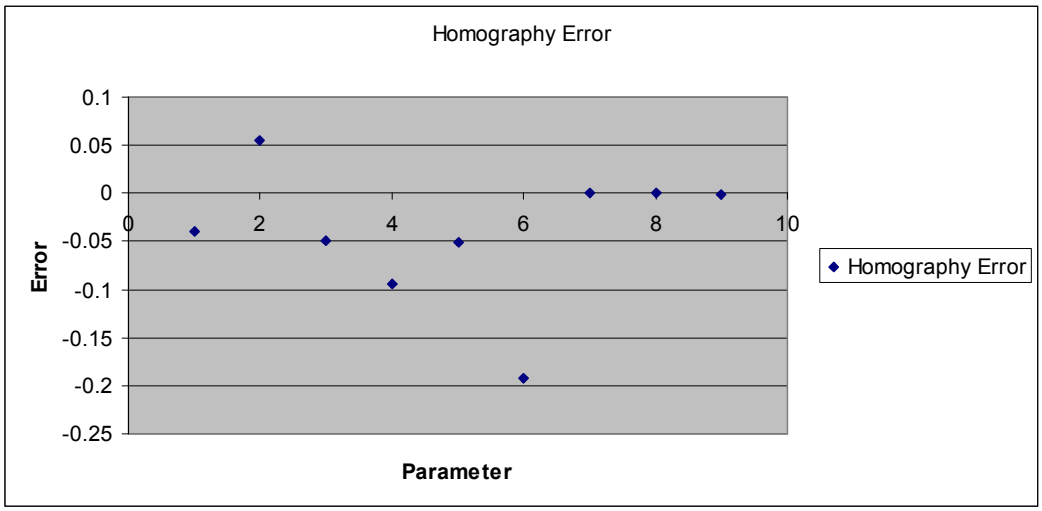


Figure 29: Homography Error for LMedS Algorithm

The next step is image transformation. It is performed using the homography information and transformed image is added to the mosaic. The mosaic results for MA3 for real and synthetic videos are given in Figure 32 and Figure 33, respectively.

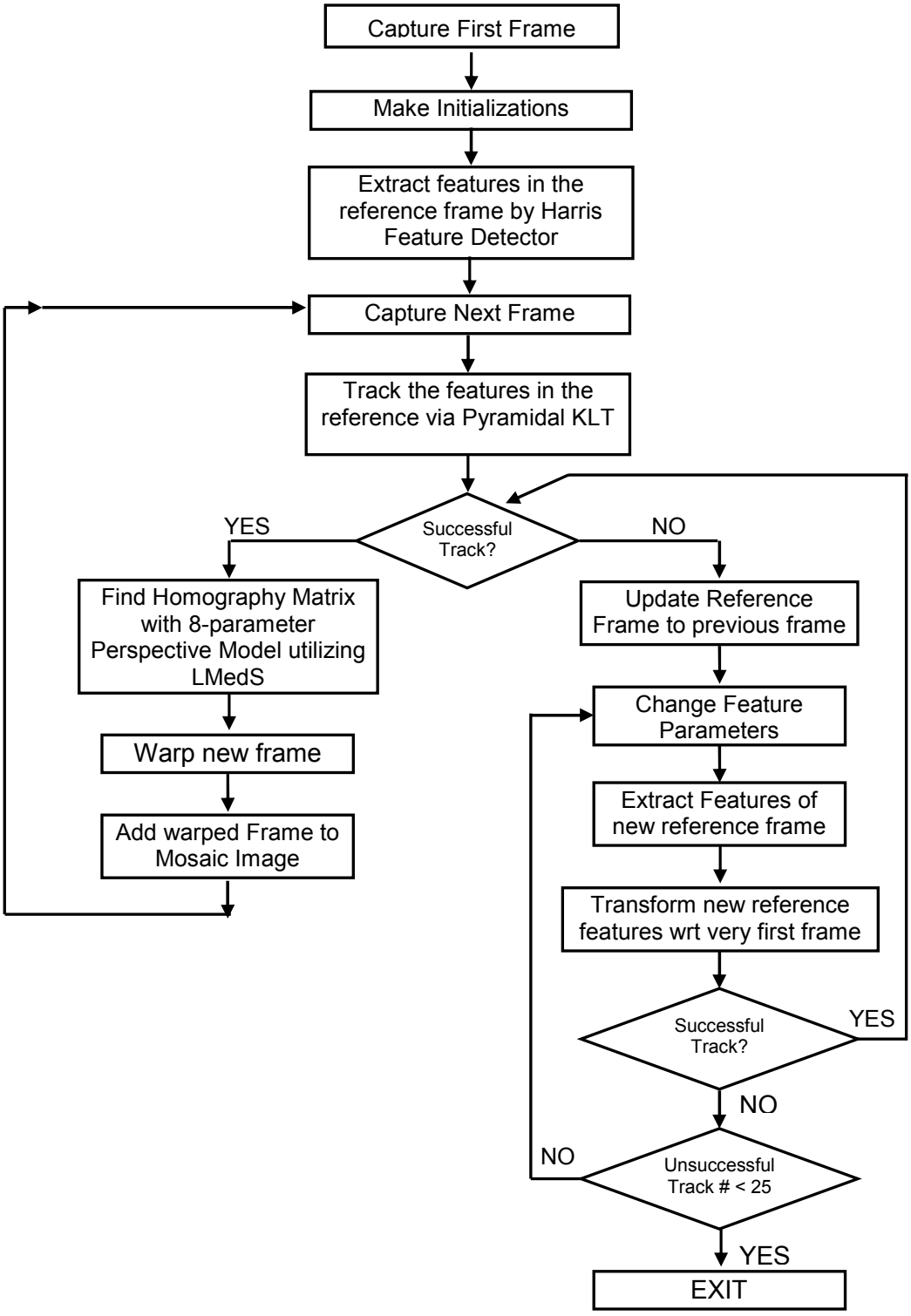


Figure 30: Flowchart of Mosaicing algorithm 3

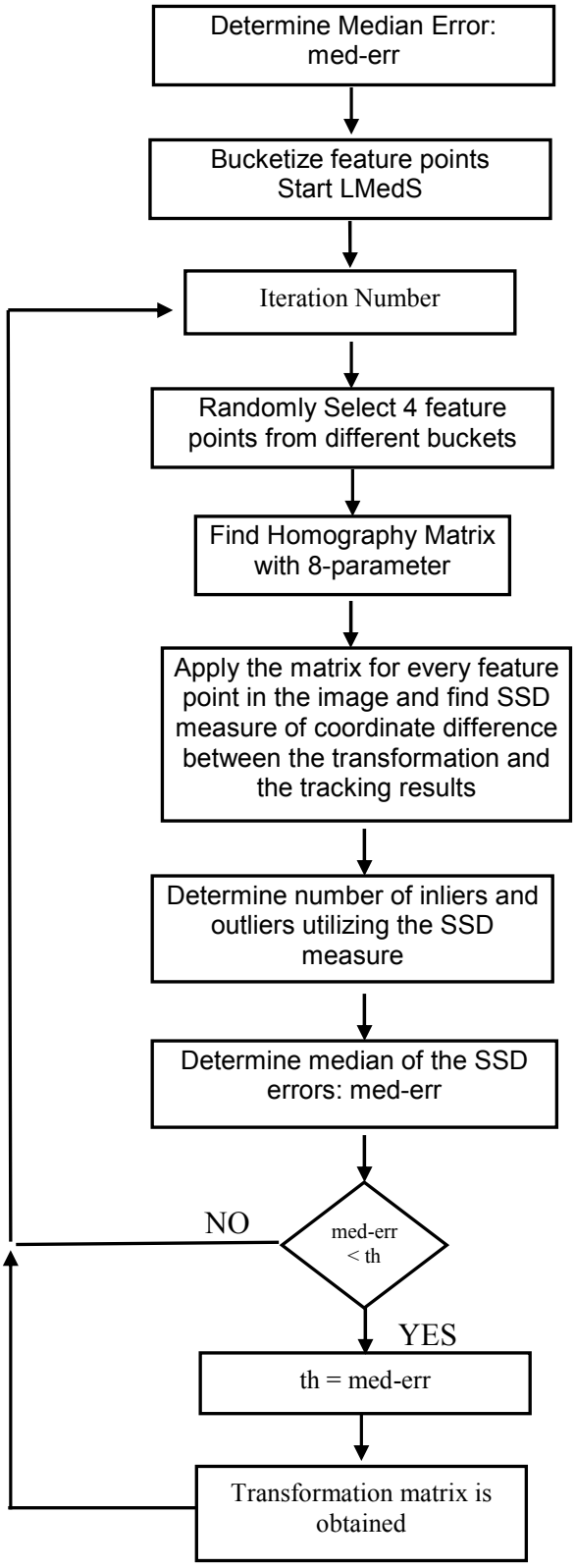
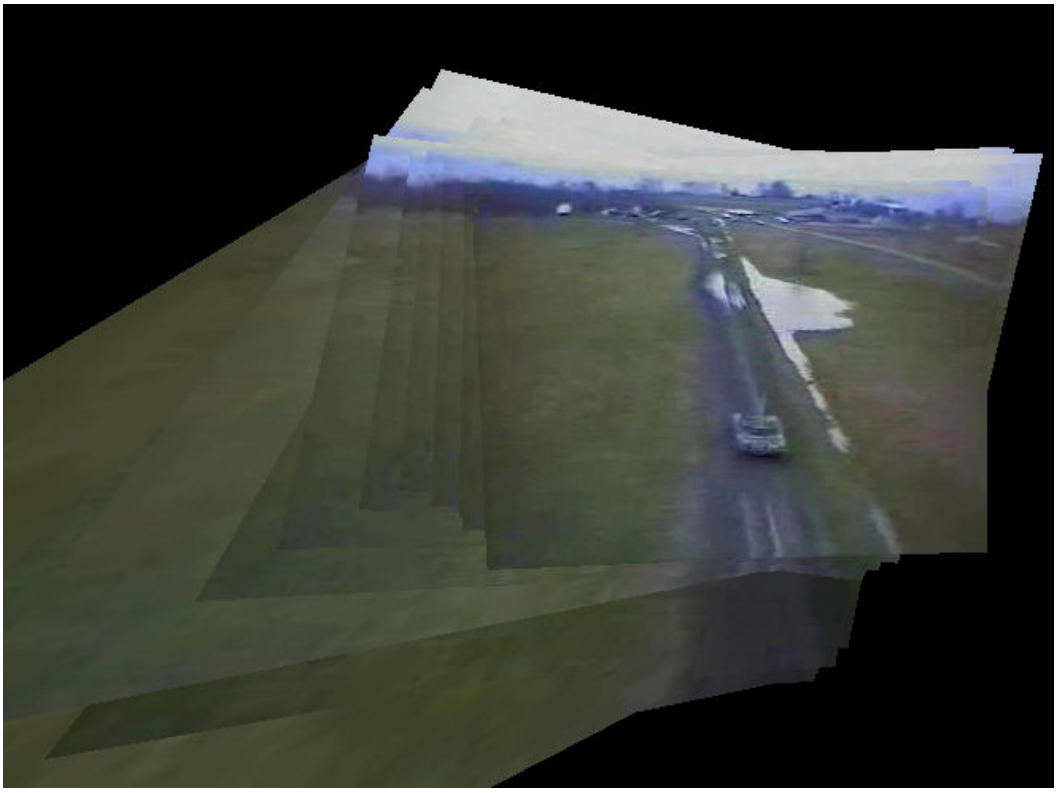


Figure 31: LMedS Implementation

(a)



(b)

Figure 32: Mosaic Results of Mosaic Algorithm 3 for real videos (a) zeppelin (b) uav videos

(a)

(b)

Figure 33: Mosaic Results of Mosaic Algorithm 3 for synthetic videos

(a) wall (b) amasra videos

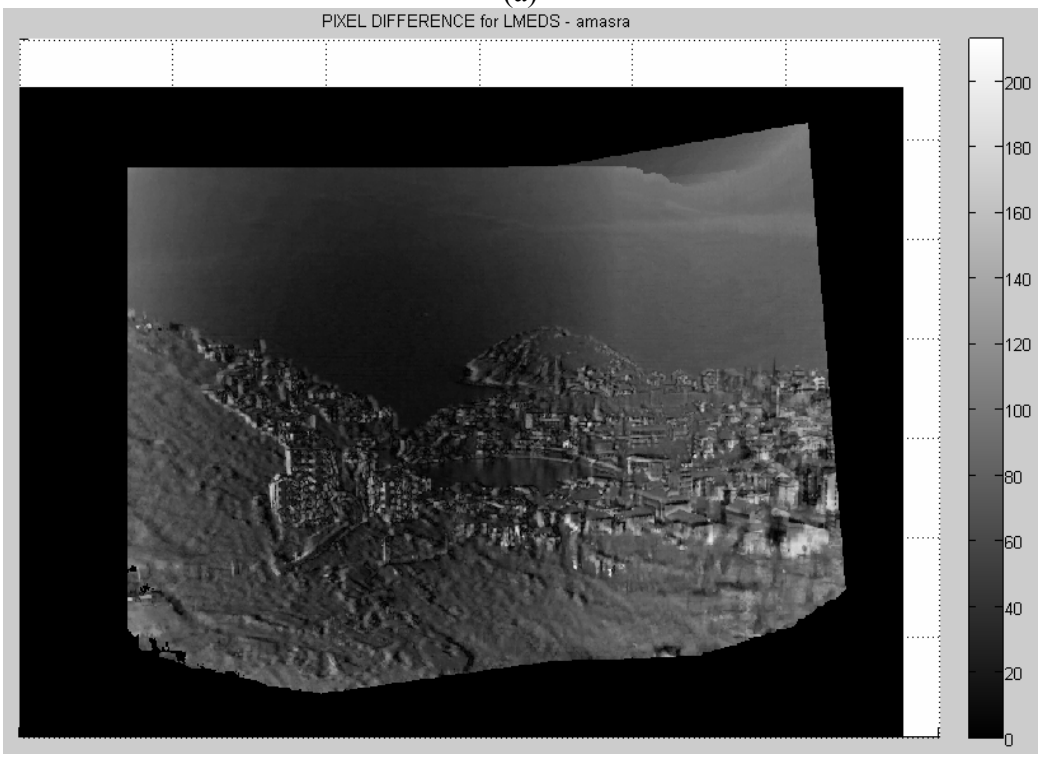
The SNR results which are obtained by Equation (42) and Equation (43) of MA2 for the synthetically generated videos are given in Table 9.

Table 9: SNR values of synthetically generated videos for MA3

	Video	
	Amasra	Wall
SNR value	29.0224	41.39

The pixel-wise intensity difference between the actual images in Figure 12 and mosaic images in Figure 33 is given in Figure 34.

(a)



(b)

Figure 34: Pixel-wise frame difference of MA3 for (a) wall (b) amasra mosaics

4.4 Mosaicing Algorithm 4

The Mosaicing Algorithm 4 (MA4) is composed of the following steps:

1. Feature extraction via SIFT key point extractor
2. Feature matching with SIFT matching
3. Homography Transformation Matrix with Least Squares

In the first three algorithms, corner like features are extracted via Harris Corner Detector and the corresponding features in the image pair are determined by KLT tracker. MA4 differs from the first three algorithms in this manner and the first two steps are implemented in a completely different way. The flowchart of the MA4 algorithm is given in Figure 35. In MA4 features are extracted by Scale Invariant Feature Transform (SIFT) algorithm. In SIFT algorithm, the points selected as keypoint have different characteristics than features of Harris Corner Detector. The feature points extracted from the reference frames of the videos are given in Figure 37. There are 4 critical parameters in SIFT feature extraction process that effects the number of features.

1. Number of sampled intervals per octave : SIFT_INTVLS
2. Image size before pyramid construction: SIFT_IMG_DBL Increase number of stable points by a factor of 4
3. Width of descriptor histogram array SIFT_DESCR_WIDTH 4
4. Number of bins per histogram in descriptor array SIFT_DESCR_HIST_BINS 8

In [6], detail analysis of the parameters of SIFT algorithm is performed and the parameters are selected accordingly. In SIFT, the image, in which the keypoints are extracted, is Gaussian blurred with a variable variance between 0 and 1. Number of the blurred images is SIFT_INTVLS and it is selected as 3. In [6], some tests are performed and it is observed that increasing the number of intervals do not increase the repeatability of the keypoints. In this application, the keypoints are found for the same size of images. So, selecting this parameter smaller increases the repeatability of the keypoints. The second critical parameter, the image size before pyramid construction, makes the initial image size double and increases the number of stable keypoints by a factor of 4. However if the counter like parts in the image is widely spread over the image, this option might be eliminated in order to increase the repeatability factor. Other two parameters in descriptor vector construction are, the gradient histogram of the selected point which is calculated around a SIFT_DESCR_WIDTH pixel and gradient vectors that are stored SIFT_DESCR_HIST_BINS directions. Using these parameters 128 dimension descriptor vector is obtained as proposed in [6].

The parameters for different videos are given in Table 10:

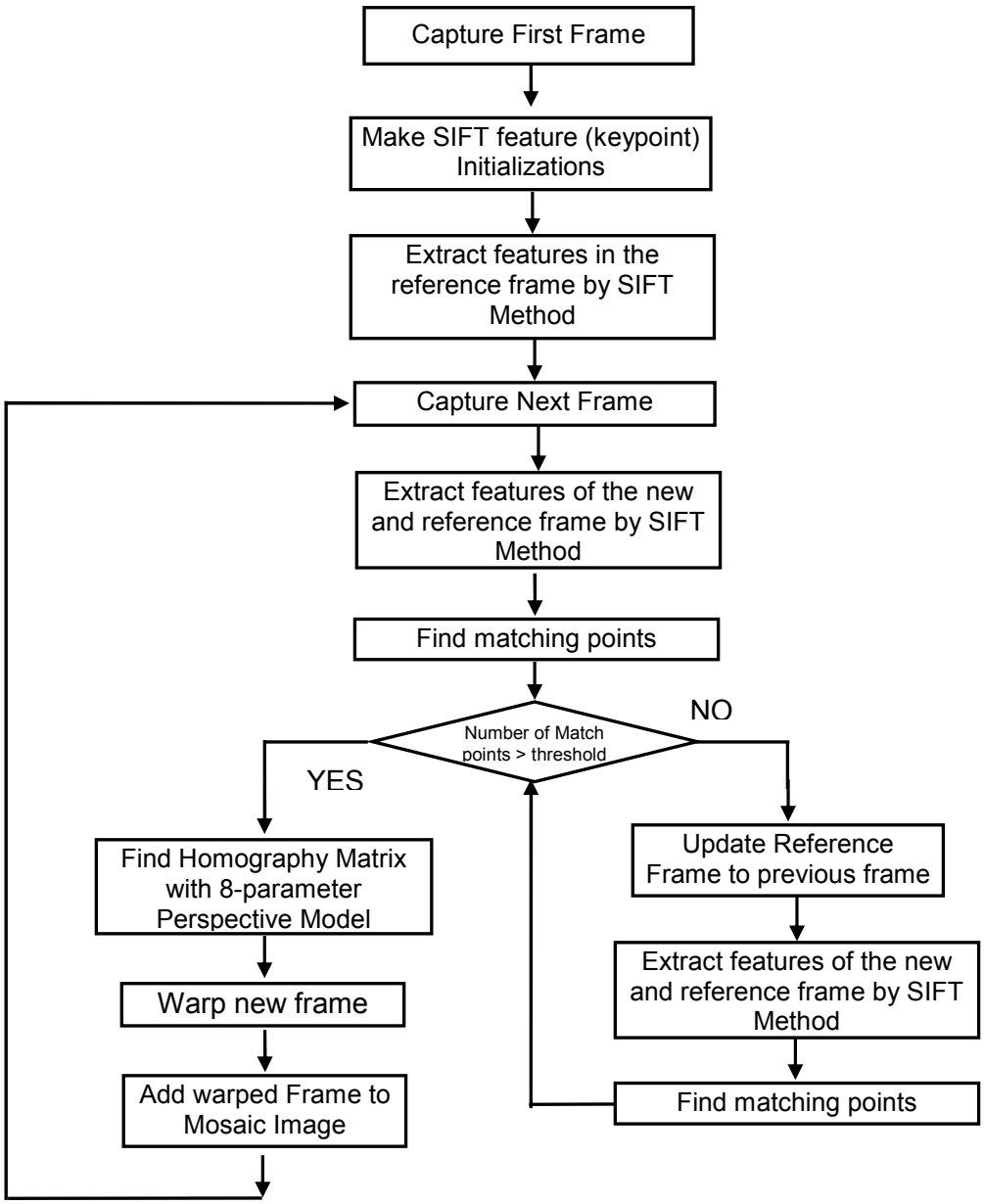
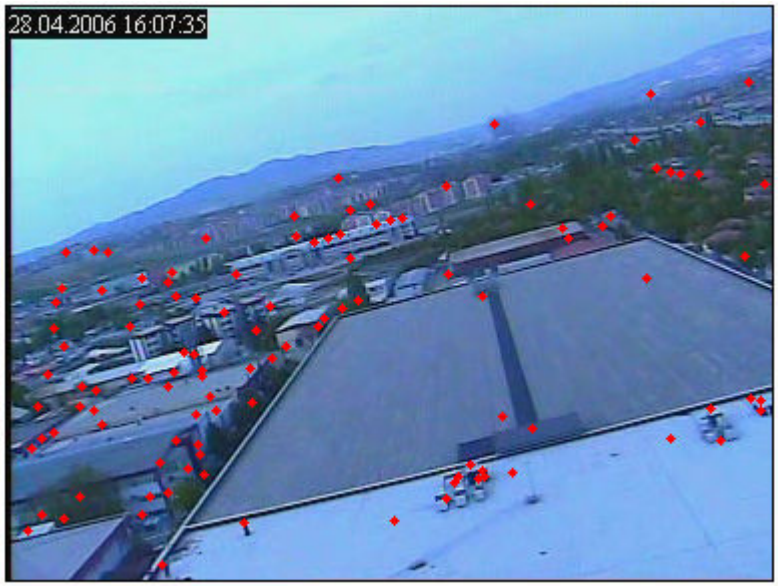


Figure 35: Flowchart of Mosaicing Algorithm 4

Table 10: SIFT Keypoint Determination Parameters (by DoG)

		VIDEO			
		Zeppelin	Uav	Wall	Amasra
PARAMETER	SIFT_INTVLS	3	3	3	3
	SIFT_IMG_DBL	1	1	1	0
	DESCR_WIDTH	4	4	4	4
	SIFT_DESCR_HIST_BINS	8	8	8	8



(a)

(b)

Figure 36: Keypoints extracted by SIFT algorithm (DoG) for real videos

(a) zeppelin (b) uav

(a)

(b)

Figure 37: Keypoints extracted by SIFT algorithm (DoG) for synthetic videos (a) wall (b) amasra

After the keypoints are selected according to criteria above and a descriptor vector is constructed for each keypoint for the given image pairs, the matching keypoints are found. Since the image pairs have some non-overlapped areas, different keypoints are extracted for each image and a matching operation is needed in order to find the correspondent keypoints. There are two critical parameters for matching operation:

1. Division threshold
2. Euclidean difference threshold

The matching operation in SIFT is performed by means of the information in keypoint descriptor vectors. One keypoint is selected in the first image and vector differences between all the 128 dimensional keypoint descriptor in the next image are calculated. During this operation the minimum and the second minimum differences are stored. Then, the keypoint that gives the minimum vector difference is chosen as the candidate pair. Then another control is performed using the division threshold. The minimum and second minimum vector differences are divided into each other. The keypoint pair is accepted if the division result is below the division threshold. This factor guarantees to discard the keypoint which have closer characteristics. In determination of this threshold, the division result and the distance for the pair are plotted and the division threshold is chosen such that the pairs that have larger distance are rejected. Figure 38 shows the relation between this division and Euclidean difference. The matching operation is very critical in SIFT application since incorrect matches are obtained from SIFT. Figure 38 shows the Euclidean difference between the

matching pairs for “zeppelin” video. This video has small displacement between the frames and there is no moving foreground in the video so, small distance is expected. However, Figure 38 shows that there are a lot of keypoints which have significant displacement between the frames. In order to reject these candidate keypoints, a line is fitted to the Euclidean difference and match points which have Euclidean difference above this line is ignored. This line differs according to the video characteristics. If the frame difference between the consecutive frames is large, this line should be shifted above which allows more displacement between the matching pairs. This elimination affected the visual performance very significantly. The parameter values used in implementations are given in Table 11. Different thresholds are used for different videos. The reason of this difference is the displacement between the frames in the video. If the displacement is large, this factor has to be selected larger. In practice, the best way of selecting the threshold is to examine the characteristics as in Figure 38 and Figure 39.

Table 11: Parameters of SIFT matching

		VIDEO			
		Zeppelin	Uav	Wall	Amasra
Parameter	Division Threshold	0.3	0.3	0.3	0.3
	Euclidean Difference Threshold	4/3	5	2	4/3
	Keypoint Match Threshold	150	45	450	400

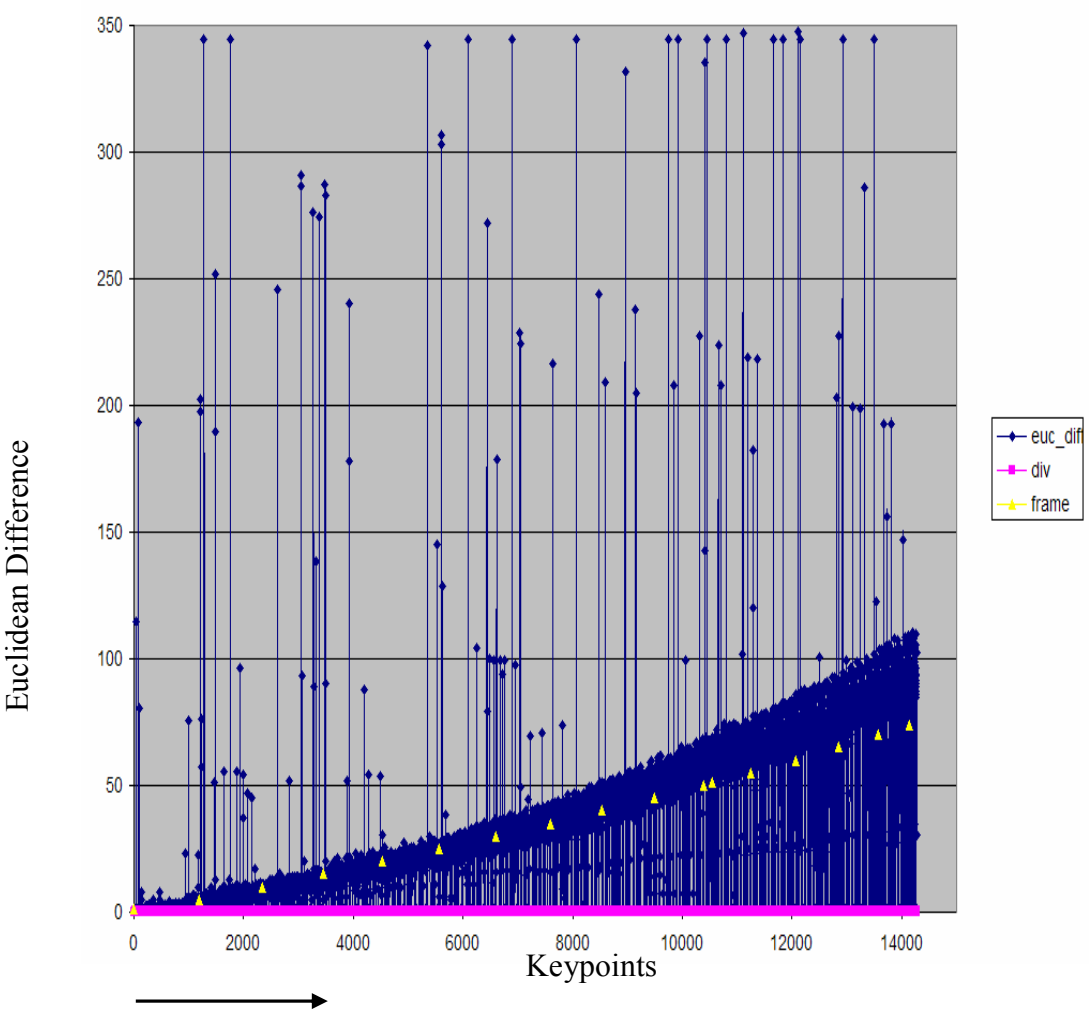


Figure 38: Distance between the matched points

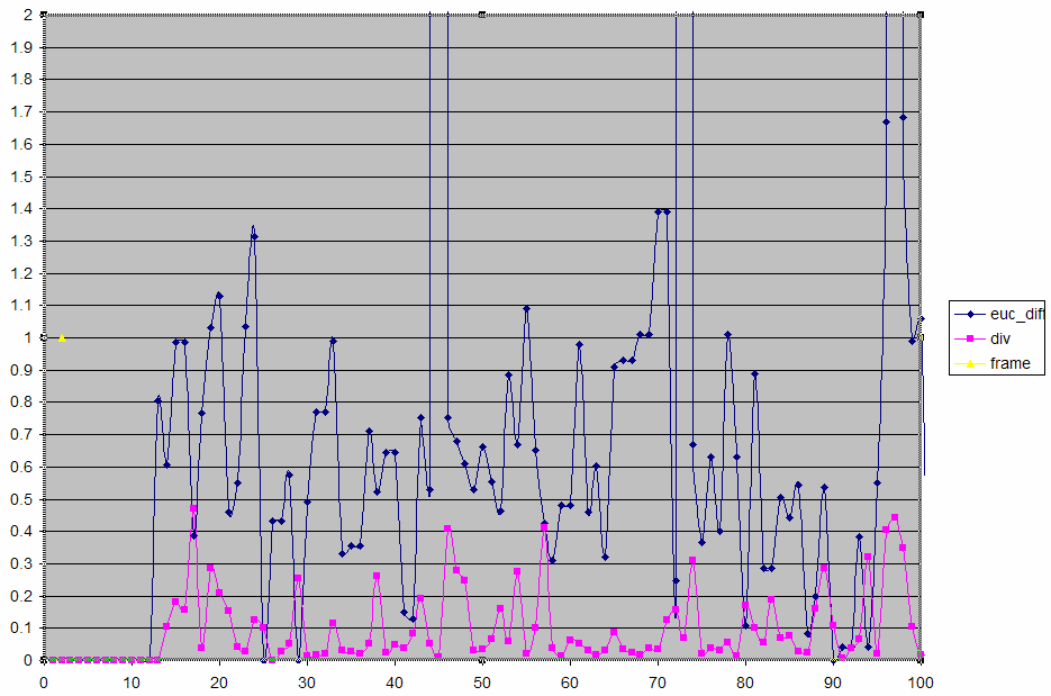


Figure 39: Detailed first portion of Figure 38

In SIFT algorithm matching operation is improved by Hough Transformation in which the keypoints are fitted to a suitable model. Then the keypoints which do not fit the model are rejected. In this study Hough Transform is not utilized since it takes very long time compared to SIFT feature Extractor and matching algorithms.

After keypoints are extracted and the correspondences are found, another control is performed. If the number of matches found is below the Keypoint match threshold, then the reference frame is updated to previous frame. Then keypoints of the previous and current frames are extracted and matches are found. If the

reference frame is updated, number of matches is not controlled in the second time and the match keypoints are stored.

After matching keypoints are determined, the homography matrix is calculated with these keypoint pairs. Then, the second image is warped according to the reference frame using the homography matrix information. When the warped image is obtained, mosaic image is updated by just adding the new pixels. The mosaic results of Mosaicing Algorithm 4 for the real and synthetically generated videos are given in Figure 40 and Figure 41, respectively.

The SNR results of the SIFT algorithm for synthetically generated “wall” and “amasra” videos are given in Table 12:

Table 12: SNR values for SIFT algorithm

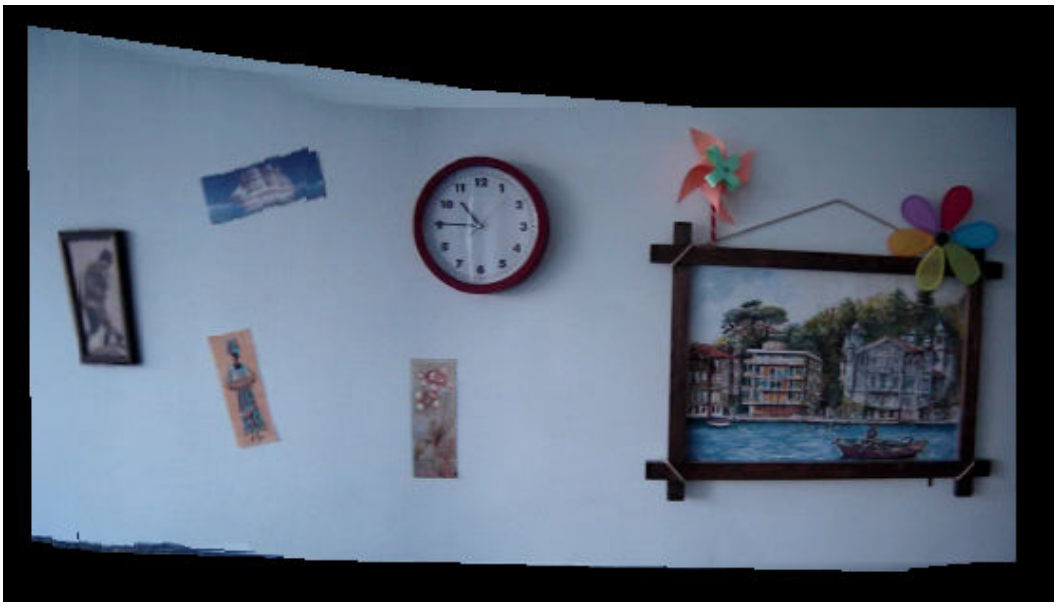
	Video	
	Amasra	Wall
SNR value	28.8987	40.73

The pixel-wise differences between the actual “wall” and “amasra” frames in Figure 12 and the mosaic result of the SIFT algorithm in Figure 41 is given in Figure 42.

(a)

(b)

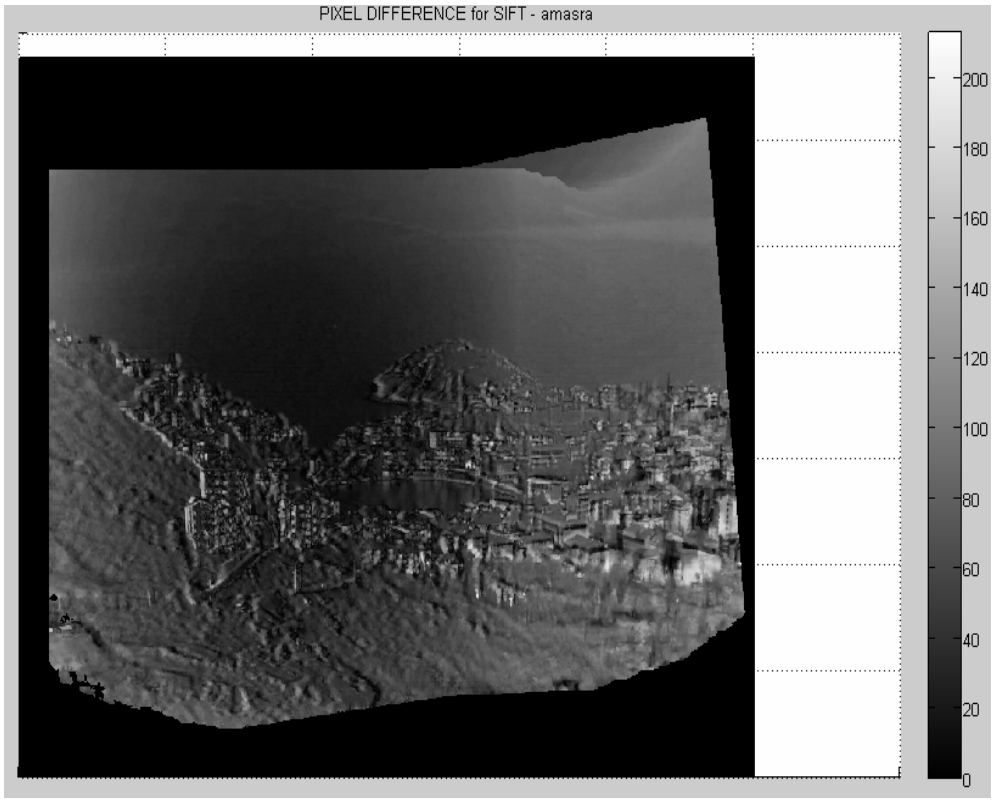
Figure 40: Mosaic Results of Mosaic Algorithm 4 for real videos (a) zeppelin (b) uav videos



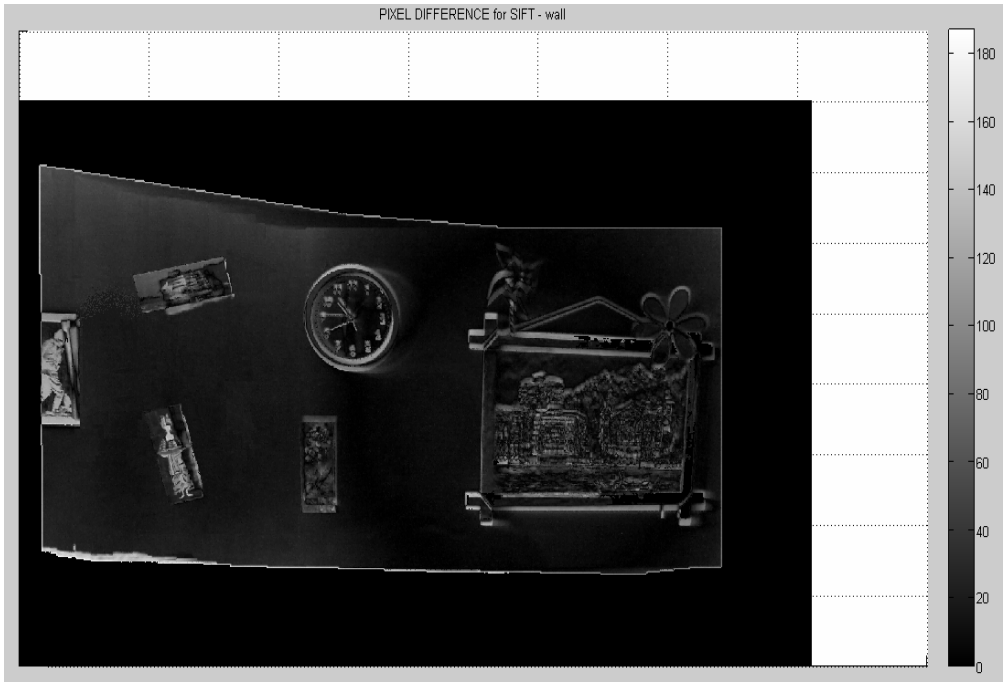
(a)

(b)

Figure 41: Mosaic Results of Mosaic Algorithm 4 for real videos (a) zeppelin (b) uav videos



(a)



(b)

Figure 42: Pixel-wise frame difference for (a) amasra (b) wall videos

The failure of the SIFT algorithm near Atatürk's picture is because of the low matching performance of the implemented algorithm. Since the video is not captured from high altitude, small displacements of the camera results large perspective difference, the relation between the Euclidean Differences between the keypoints' coordinates can not be fitted to a line. So rejection of the incorrect keypoints can not be performed efficiently. Besides, due to the fast perspective changes, update of the reference frame has to be performed frequently. Because of these reasons the perspective is distorted resulting bad visual performance especially for the new edge points.

The reason of the low SNR value of the wall image is the existence of an empty wall in the background. Since the intensity of the wall is almost constant, SNR value is not affected from incorrect warping. If the edges of the pictures are paid attention, it can easily be recognized that SIFT gives worse results for "wall" video which are compatible with the above paragraph.

4.5 Comparison of the Algorithms

In this section, performances of the algorithms with respect to SNR and time complexity are compared.

Considering the SNR values in Table 13 and pixel wise intensity differences of the "amasra" and "wall" mosaics, it is observed that MA2 gives the best result.

The visual performance of MA2 algorithm is better than other first two algorithms which differ only in Homography matrix calculation process. Compared to MA1 this result is expected since outlier elimination is performed in MA2 than Least Squares. Outlier rejection is performed in also MA3. The results reveal that outlier elimination for MA2 is better than MA3 method. Actually this result is expected since MA3 considers only median of the differences although MA2 deals with all feature coordinate differences. This result also derived from the homography parameter errors given in Figure 19 for Least Squares, Figure 23 for MA2 and Figure 29 for MA3 methods. The SNR value of the MA4 method is about the same for the other three for "amasra" and it is lower for wall video. The reason of this low value is that the keypoints of the MA4 are located in the right painting which causes nonuniform distribution of the features. So perspective can not be obtained accurate enough resulting a lower SNR value and distortion in the image as seen in Atatürk's painting.

Table 13: SNR Values of the algorithms

		Video	
		Amasra	Wall
SNR	MA1	27.5606	41.42
	MA2	29.52	41.69
	MA3	29.0224	41.39
	MA4	28.8987	40.73

The time complexities and consumed times of the algorithms are given in Table 14 and

Table 15, respectively. The complexity calculation is performed for the major steps in the Table 14. In feature extraction step, MA4 is the most time consuming since it utilizes DoG Method in SIFT. On the other hand, MA3 has the highest complexity in homography matrix calculation. For these reasons, these algorithms are not suitable for real time applications. On the other hand, MA1 and MA2 methods seem comparable when the spent times are taken into account. However it can easily be recognized from Table 14 that the time spent by MA2 increases rapidly as the image size and feature number increase.

In MA2, the iteration number in homography matrix calculation (RANSAC) step is updated in each step. The closeness of MA1 and MA2 values indicate that iteration number is updated very rapidly which only occur if the number of outliers is very limited. So for video in which the outliers are significant, MA2 consumes much more time and is not again suitable for real time purposes.

Table 14: Time complexity of the algorithms

		Algorithms			
		MA1	MA2	MA3	MA4
Complexity of Algorithm Steps	Feature Extractor	n^2	n^2	n^2	n^4
	Feature Tracking/Matching	n	n	n	n^2
	Homography Matrix Calculation	n	$n^3 + \log(n)$ (max)	n^5	n
	Warping	n^2	n^2	n^2	n^2
Complexity of the Algorithms O(n)		n^2	n^3	n^5	n^4

Table 15: Spent time for algorithms for two different frames

		Algorithms			
		MA1	MA2	MA3	MA4
Time	Frame1	1.03	1.04	14.06	5.11
	Frame50	1.05	1.12	14.01	5.18

Table 16: Spent time for algorithms for two different frames_ Consecutive Homography

		Algorithms		
		MA1	MA2	MA3
Time (sec)	Frame1	0.39	0.4	6.03
	Frame50	0.4	0.41	6.21

In the point of view of SNR value, MA2 gives the best result whereas the MA1 algorithm is the best for time complexity. Besides, if accurate results are needed with no time restriction, full SIFT algorithm should be tried.

In situations where both accurate and fast mosaic construction is required more information is needed which can be supplied either by some pre-processing or as input. This result can be achieved by implementing MA1 with providing extra information.

CHAPTER 5

SUMMARY AND CONCLUSIONS

In this study, the problem of mosaic image construction is addressed and different algorithms in the literature are implemented and compared. The comparison criteria of these algorithms are visual performance (SNR), computational load and time complexity.

First detailed information about the sub-algorithms has been presented. The feature extraction algorithms serve to extract reliable and distinctive interest points which make the tracking/matching more stable. Features are extracted using Minimum Eigenvalue Method, Harris corner Detector and DoG (SIFT) feature extraction algorithms. Among these features, it is observed that Harris Corner Detector gives the most reliable points. These features are the inputs for tracking/matching module in which optical flow based pyramidal tracking or SIFT matching algorithms are exploited. The algorithms are optimized for the properties of the video inputs in which the features extracted serves the

requirements of tracking algorithms. The tracking algorithm utilizes the iterative pyramidal KLT, which is a well known tool in computer vision. Pyramid structure is preferred to obtain more consistent and reliable results.

After feature extraction and tracking/matching operations, homography transformation is calculated in order to handle camera motions and adjust the perspective distortions. This homography transformation is utilized by three different algorithms. The first one is “Least Squares” method followed by an SSD minimization. In situations where more robust results are acquired, the other two methods, RANSAC and LMedS, which are computational expensive algorithms, are preferred. Among these three algorithms, it is observed that RANSAC gives the most accurate results from visual point of view. Then, the reliable and robust transformation parameters are utilized for warping process and final mosaic image is obtained after adding of the warped image.

In the final step, the performances of the given algorithms on the overall mosaicing system are presented in SNR and time complexity point of views. As shown in the results, MA2 gives the best SNR performance among the other algorithms. The reason is that MA2 eliminates the outlier features in a more intelligent manner when compared with MA1 and MA3. Besides, MA4, which uses different techniques for both feature extraction and matching algorithms, gives the worst results among all. Considering time complexity, MA1 has the best performance whereas MA3 has the worst. This statement reveals that homography calculation quite affects the time complexity of the overall

algorithm. Additionally, feature extraction step in MA4 is a time consumer step compared to Harris Corner Detector in MA1, MA2 and MA3.

To conclude, the decision of the algorithm for aerial video mosaicing has to be given with respect to system requirements which determine the time and visual quality thresholds. The performance of MA1 algorithm is optimum for the systems which require real time and compatible visual performance and can be improved by adding small intelligence on the characteristics of the system.

5.1 Future Work

As a future work, we will perform some improvements to increase the performance of the mosaic systems. These improvements include brightness compensation, real time working and SNR value enhancements. Brightness compensation will be performed in order to improve visual performance and to increase the stability of tracking/matching performances.

In order to decrease the time complexity some improvements including the information of the video and system characteristics will be added to make the algorithm work in real time. Also, the mosaic results will be improved using low resolution satellite images as mentioned in [17] which can make the system work faster.

In high resolution applications, the SNR values should be increased in order to increase the visual performance. The effect of robust feature selection will be examined with the new method in [16] published in 2007 which improves feature quality of Harris Corner Detector. Besides, the effect of increasing the matching performance to the SNR value for the SIFT matching will be done as proposed in [6].

CHAPTER 6

REFERENCES

[1] M. Ramachandran and R. Chellappa. “Stabilization and mosaicing of airborne videos”, ICIP06, pp.345-348, 2006.

[2] M. Irani and P. Anandan, “About direct methods”, Vision Algorithms '99, pp. 267-277, 1999.

[3] J. Shi and C. Tomasi, “Good features to track", Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn., pp. 593-600, 1994.

[4] C. Harris and M.J. Stephens, “A combined corner and edge detector”, Alvey Vision Conference, pp. 147–152, 1988.

[5] P.Meer, D. Mintz, D.Y. Kim and A. Rosenfeld, “Robust regression methods in computer vision: A review”, International journal of Computer Vision, pp. 59-70, 1991.

[6] D.G. Lowe, “Distinctive image features from scale-invariant keypoints”, International Journal of Computer Vision, vol. 33, no. 2, pp. 91-110, 2004.

[7] M.A. Fishler and R.C. Bolles,“Randon sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography”, Comm. ACM, vol. 24, pp. 381-395, 1981.

[8] A. Bartoli, N. Dalal, B. Bose and R. Horaud, “ From video sequences to motion panoramas” , IEEE Workshop on Motion and Video Computing, pp. 201-207, Dec. 2002.

[9] C. Tomasi and T. Kanade, “Detection and tracking of point features

technical report", Carnegie. Mellon University Technical Report CMU-CS-91-132, Apr. 1991.

[10] J. Y. Bouguet, "Pyramidal implementation of the Lucas Kanade feature tracker description of the algorithm", Intel Corporation Microprocessor Research Labs, 2000.

[11] M. Irani and P. Anandan. "About direct methods. In B. Triggs, A. Zisserman, and R. Szeliski, editors, Vision Algorithms: Theory and Practice", pp. 267–277,July 1999.

[12] R.C. Hardie, T. Tuinstra, K. J. Barnard, J. G. Bognar, and E. A. Armstrong,
"High resolution image reconstruction from digital video with in-scene motion", ICIP97, 1997.

[13] M. Irani, P. Anandan, and S. Hsu, "Mosaic based representations of video sequences and their applications", In Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA, pp. 605–611, June 1995.

[14] A. Smolic and T. Wiegand, "High-resolution video mosaicing", ICIP01, pp. 872-875, 2001.

[15] P. H. S. Torr and A. Zisserman, "Feature based methods for structure and motion estimation. In W. Triggs, A. Zisserman, and R. Szeliski, editors, Vision Algorithms: Theory and Practice", LNCS, pp. 278–295, July 1999.

[16] S. C. Pei and J. J. Ding, "Improved Harris' algorithm for corner and edge detections", ICIP07, vol. 3, pp. 57-60, 2007.

[17] Y. Lin and G. G. Medioni, "Map-enhanced uav image sequence registration", IEEE Workshop on Applications of Computer Vision, 2007.

[18] H. Moravec, "Obstacle avoidance and navigation in the Real World by a Seeing Robot Rover", Carnegie-Mellon University Robotics Institute Technical Report CMU-RI-TR-3, Sep. 1980.