TARGET FOLLOWING CAMERA SYSTEM BASED ON REAL TIME RECOGNITION AND TRACKING

A Thesis Submitted in Partial Fulfillment

of the Requirements for the Award of the Degree of

Master of Technology

In

Signal and Image Processing

by

R.V.YASWANTH KUMAR

Roll No: 212EC6188

Department of Electronics & Communication Engineering

National Institute of Technology, Rourkela

Odisha- 769008, India

May 2014

TARGET FOLLOWING CAMERA SYSTEM BASED ON REAL TIME RECOGNITION AND TRACKING

A Thesis Submitted in Partial Fulfillment

of the Requirements for the Award of the Degree of

Master of Technology
In
Signal and Image Processing

by

R.V.YASWANTH KUMAR
Roll No: 212EC6188

Under the Supervision of

PROF. K. K. MAHAPATRA

Department of Electronics & Communication Engineering

National Institute of Technology, Rourkela

Odisha- 769008, India

May 2014

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY, ROURKELA

ODISHA, INDIA - 769008

CERTIFICATE

This is to certify that the thesis entitled **"TARGET FOLLOWING CAMERA SYSTEM BASED ON REAL TIME RECOGNITION AND TRACKING"** submitted by **Mr. R. V. YASWANTH KUMAR** bearing roll no. **212EC6188** in partial fulfillment of the requirements for the award of Master of Technology in Electronics and Communication Engineering with specialization in **"Signal and Image Processing"** during session 2012-14 at National Institute of Technology, Rourkela, is an authentic work carried out by him under my supervision and guidance.

To the best of my knowledge, the matter embodied in the thesis has not been submitted to any other University / Institute for the award of any Degree or Diploma.

Prof. Kamala Kanta Mahapatra
Dept. of ECE,
National Institute of Technology,
Rourkela

Date:

Dedicated to My Family, Teachers and Friends

ACKNOWLEDGEMENTS

First of all, I would like to express my deep sense of respect and gratitude towards my advisor and guide **Prof. K.K. Mahapatra**, who has been the prime force which helped me successfully accomplish this work. I feel immensely honored to have his guidance, for his constant encouragement, invaluable advice and values which helped me boost my confidence time and again. His wisdom and optimism have had an invaluable influence on my career and propel me to have a better perspective and outlook towards the future. It was a wonderful opportunity to work with such a wonderful person.

Next, I express my humble respects to Prof. S. Meher, Prof. A.K. Swain, Prof. L.P.Roy, Prof. A.K.Sahoo, Prof. D.P. Acharya, Prof. S.K. Patra, Prof. U.K.Sahoo, Prof. Samit Ari and Prof. Poonam Singh for always being there to clarify my silliest doubts. Their knowledge has been a great treasure for which it always inspired me to cross all the hurdles I encountered and for that I thank them from the bottom of my heart.

I would also like to thank all my friends, especially Abhinav Karthik, for all the thought provoking and knowledge sharing discussions that we had, which stimulated me to see beyond the obvious. I have enjoyed their warmth and friendliness all through my stay at NIT, Rourkela.

I am especially indebted to my parents as they were always there for me in all the ups and downs of my life. They are and will be my constant support throughout my life. They were my first teachers and always will set great examples for me live, study, and work.

R.V. Yaswanth Kumar
Roll No: 212EC6188
Dept. of ECE
NIT, Rourkela

Date:
Place:

ABSTRACT

A real-time moving target following camera system is presented in this study. The motion of the camera is controlled based on the real-time recognition and tracking of the target object. Scale Invariant Feature Transform (SIFT) based recognition system and Kanade-Lucas-Tomasi (KLT) tracker based tracking system is presented to recognize and track the moving target. SIFT algorithm is slow but efficient in recognizing the objects even though they undergone some affine transformations. KLT tracker algorithm is simple and has reduced computations, hence improves the tracking performance. The analysis is performed in hardware which consists of a camera mounted on a two servo motor setup, one for pan and other for tilt, and an Arduino board capable of handling the movement of two servo motors. As there is hardware implementation, a computationally simplified technique is employed. Since both SIFT and KLT tracker are feature based techniques, we pass the features extracted by SIFT to KLT tracker for simplifying the process. The recognition and tracking tasks are performed in PC and the PWM signals are generated accordingly and sent to servo motors through Arduino. The proposed algorithm is able to track objects even in its absence for a certain while.

Keywords− SIFT; KLT tracker; Recognition; Tracking

Table of Contents

Table of Figures

Chapter 1

Introduction to Recognition and tracking

1.1 Overview

Recognition and tracking of an object is a basic objective of Computer Vision applications. Some of the examples where the object recognition and tracking required are airport safety, road traffic control, surveillance systems, etc. Recognizing an object can be used to provide a three-dimensional interpretation of the object and in discrimination among different objects. Tracking is a vast topic, with extensive research being done currently. Object tracking is estimating the motion or path of the object of interest within the image as it moves around a scene.

The main problem of object recognition is the ubiquitous problem of Computer Vision. For a given 3-dimensional scene data and for a given object model set, the primary goal of obect recognition is to highlight the location of the objects within the image. The key issues to achieve it are sensor data acquisition, object model representation, data matching, and model matching. In last years, these issues have been analyzed with different techniques. Some of them rely on the mathematical models of the object being recognized. Its main goal is to estimate the object of interest's location and orientation e.g., acronym [12] or the approach proposed by Fua [13]. The other category of recognition involves cues rather than shape, e.g., MSYS [14] and Condor [15]. The literature survey of techniques of object recognition is presented in [16]. There are two categories of tracking techniques: recognition-based [17] and motion-based tracking techniques [18]. The task of tracking in recognition-based tracking techniques is performed by recognizing

the object and extracting its position in successive images. The advantage of this type of tracking is that it can be achieved in three dimensions and one more is that the object translation and rotation can be estimated. The disadvantage is that the high computational complexity limits the tracking performance. Motion based tracking tracks the object by estimating the motion parameters of the object. The disadvantage is that they will track any moving object regardless of shape.

It is difficult to integrate recognition and tracking tasks in common spatiotemporal space, because of occlusions and noise that can generate false object detection in the scene. Other thing in difficulty is that both the tasks are based on different kind of features which increases computational complexity of whole system. Koller et al. [20] proposed recognition based tracing system for traffic scenes. The system uses priori knowledge of shapes and motion of objects to be recognized and tracked. Parameterized vehicle model considers shadow edges also which made the system efficient in complex lighting conditions. The disadvantage of this method is computational complexity. Malik et al [21] also proposed the similar type of system that overcomes the problem of occlusion in multiple objects' tracking by using contour type tracking algorithm. The position and motion of object contours are estimated by linear Kalman filter. But the recognition is limited to 2-D object shapes to resolve the problem of tracking overlapped and adjacent objects. Foresti [22] proposed a vision based surveillance system to monitor the railway crossings from a distance. The system is capable of detecting, localizing, tracking and classifying multiple objects roaming in the surveillance region. To meet real-time constraints, object recognition and tracking must be separately done on different features, i.e., the spectrum [23].

1.2 Challenges

Imagine that there is a scene, in that

- Light: Light may come from one direction or from all source of directions
- Scene contrast: There may be another light which may change the visual appearance like low contrast with low light
- Object cover: Object may have different covers like glossy which makes very much difference in tracking
- Object specularity: There may be mirror like things that creates object's mirror part and this mirror part will move like independent object hiding the real object
- Object transparency: The appearance of the object may changes to what is behind the object as it looks through the object
- Object shape: There may be a variation in object's shape
- Scene clutter: There may be difficulty in distinguish the object from other things in the scene
- Scene confusion: It may be confuse with people like marching band, they all are different but looks similar in group
- Occlusion: When things partially disappear or out of site temporarily
- Moving camera: There may be movement in camera while recording

All of these circumstances need to be captured by a good recognition and tracking algorithm. So, from 30 year onwards extensive research is going on in this topic.

10 aspects and their hard cases:

Light	Disco light
Object surface cover	Person redressing
Object specularity	Mirror transport
Object transparency	Glass ball rolling
Object shape	Swimming octopus
Scene clutter	Camouflage
scene confusion	Herd of cows
Scene low contrast	White bear on snow
Scene occlusion	Object getting out of scene
Length of sequence	Return of past appearance

Table 1.1 List of challenges in tracking

1.3 Literature Review

3.1 Normalized Cross Correlation: Simplest of all algorithms. Take a target having a template of intensity values, perform the template matching everywhere in the image using normalized cross correlation. Where there is least difference between template and the new image there we declare the track would be. Note that, in this method, there is no updating of the target.

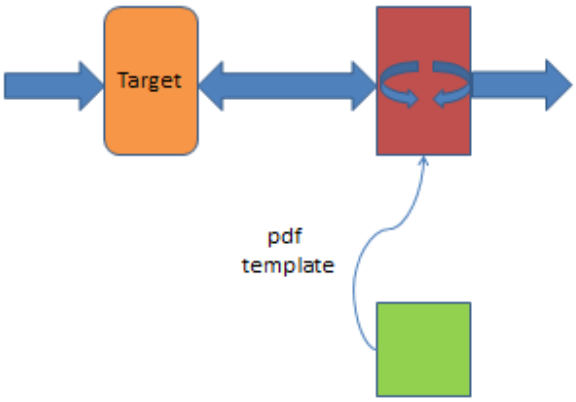


Figure 1.1: Normalized Cross Correlation

1.3.2 Lucas Kanade Tracker [31]: This is a famous tracker way a head of all algorithms in its time. It performs the matching operation same as previous technique but it allows affine transformed match between target and candidate. It employs spatiotemporal derivatives.

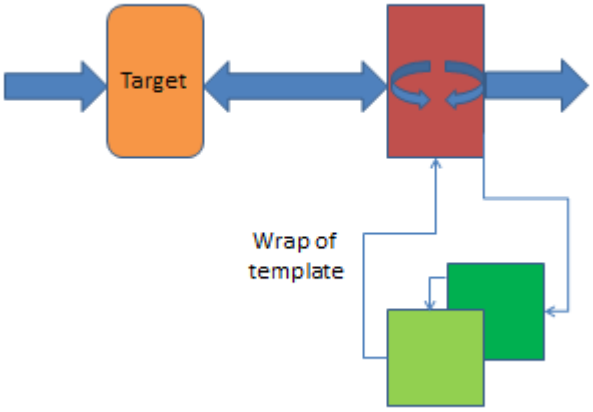


Figure 1.2 Lucas Kanade Tracker

1.3.3 Mean Shift Tracking [24]: Rather than doing the direct match of the template on the image, it does so via features derive from the image. It takes color (RGB) histogram from template and performs matching for locally taken color histogram of the image. This makes the technique independent of spatiotemporal transformations.

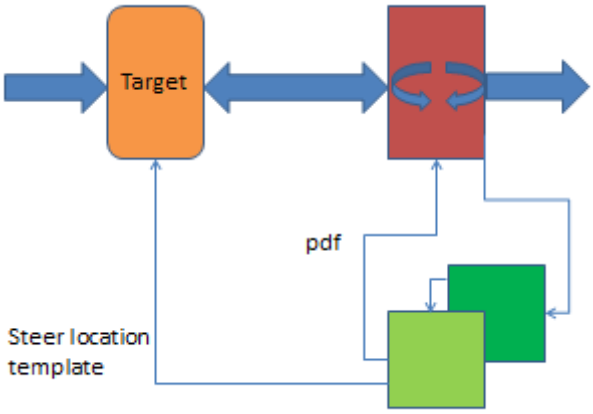


Figure 1.3 Mean Shift Tracking

1.3.4 Incremental Visual Tracking [25]: It is an extended appearance model. It captures the past in Eigen images and stores in leaking memory. So that it is easy to match with future images. It forms a matrix of all images with zero mean and it computes the set of Eigen images and then it updates them with sequential KL algorithm.

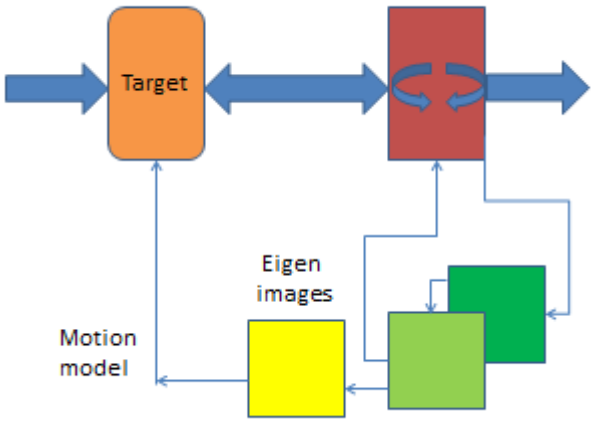


Figure 1.4 Incremental Visual Tracking

1.3.5 Foreground-Background Tracker [27]: This tracker employs different mechanism from matching. The main basic concept in this method is discrimination of foreground and background parts in the image. This concept helps when the target changes its appearance

completely. Since, we have the knowledge of background still we'll be able to track the target. The tracker trains a linear discrimination classifier. It extracts the SURF texture features from target against the local background.

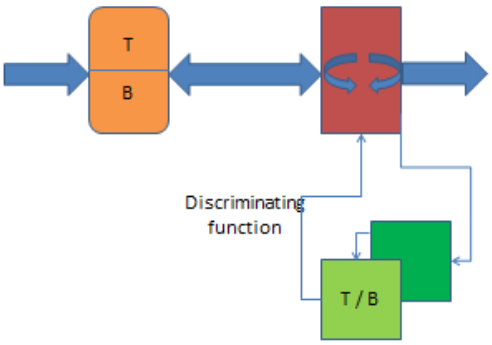


Figure 1.5 Foreground-Background Tracker

1.3.6 Tracking, Learning and Detection [29]: This is a modern technique. It searches for target everywhere in the image. It detects the target by computing Local Binary Patterns. With the help of cross correlation it will be capable of indicating where approximately the object is. It selects the one which is closest to the possibilities, made by cross correlation, moving anywhere in the image.

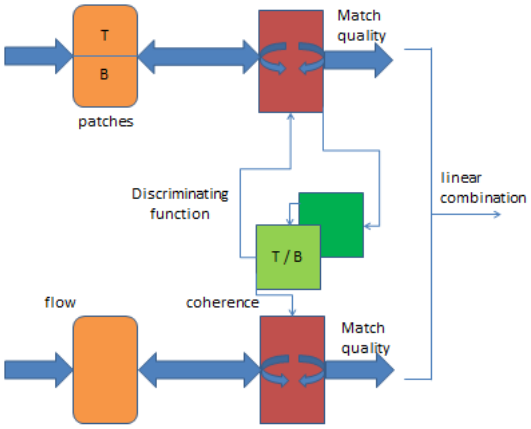


Figure 1.6 Tracking, Learning and Detection

1.3.7 Structured Output Tracking [30]: Rather than using an ordinary classifier it uses a structured classifier for foreground background discrimination, which tries to predict the translation directly. The window is described by Haar features with 2 scales.

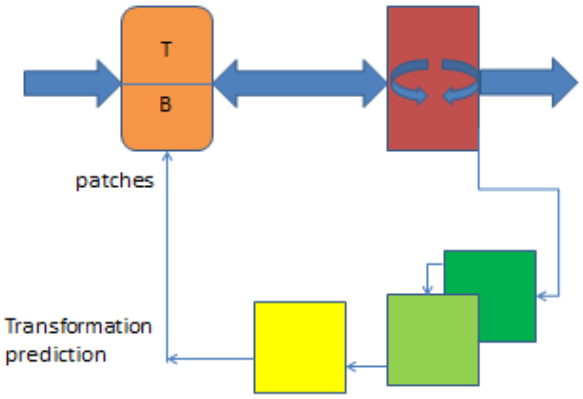


Figure 1.7 Structured Output Tracking

1.4 Systematic overview of trackers:

Trackers have to find an integral solution on the basis of:

(a) A spatial representation

(b) An appearance representation

(c) A motion model

(d) An inference method to get to the next state

(e) a method for updating the internal models

1.5 Organization of thesis:

In the remaining part of the thesis, our discussion is organized as follows. Chapter 2 explains 'scale invariant feature transform' algorithm which is used for object recognition task in this project. In this chapter, we also discuss the stepwise recognition approach we have followed.

Chapter 3 explains the ‘Kanade Lucas Tomasi tracker’ and ‘Optical Flow’, the chapter also presents the stepwise execution of the object recognition and tracking task and the corresponding results. Hardware implementation of the task and the experimental results are mentioned in chapter 4. Finally, chapter 5 concludes the thesis and gives suggestions for future work.

1.6 Conclusion:

Target aims to learn a target from the first few pictures. The target and the background may be dynamic in appearance, with unpredicted motion, and in difficult scenes. Therefore, trackers tend to be under-evaluated; they tend to specialize in certain type of conditions. Most modern trackers have a hard time beating the oldies. There is no dominant strategy found yet, apart from simplicity. That is always successful.

Chapter 2

Scale Invariant Feature Transform

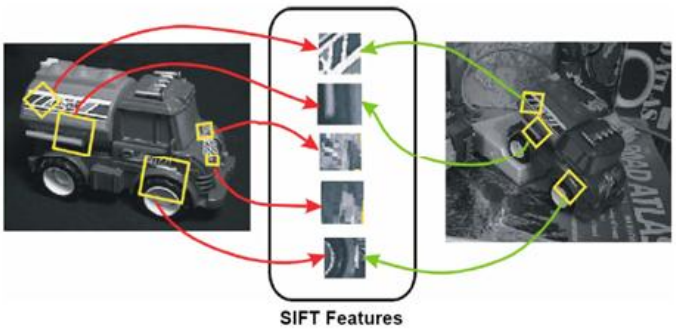
2.1 Introduction

Scale Invariant Feature Transform (SIFT) is interest point detector like Harris point detector. SIFT is proposed by David Lowe, patented by University of British Colombia. SIFT works similar to the one used in primate visual system. It converts the data of the image into coordinates which are invariant to scale. SIFT interest points are scale invariant whereas Harris point detector is not scale invariant. When we have two images of same scene taken at different scales or different depths, if we want to match the images it will be difficult. Since, SIFT provide keypoints that are invariant to scale, the image matching become possible under scale variant context. From the SIFT keypoints we extract features or descriptors which are invariant to scale, rotation and partially invariant to illumination changes. They are perfectly localized in both spatial domain and frequency domain.

Our goal is to extract distinct invariant features, so that we match features from one image to other image. We make them invariant to image scale and rotation and robust to affine distortion, change in 3D viewpoint, addition of noise and illumination changes.

Advantages:

- Locality
- Distinctiveness
- Quantity
- Efficiency



Example of SIFT Feature matching

Steps involved in feature extraction:

1. **Scale-space peak detection:** In this section, search for interest points which are either maxima or minima among of its 26 neighboring points. For this we down sample the image, blur the image and subtract the consequent image scales.

2. **Key point localization:** A proper procedure is there to determine location and scale at each key point location. Selection of key point is based on their stability.

3. **Orientation assignment:** On the basis of local image gradient directions, each key point location is assigned with one or more orientations**.** In order to provide invariance to affine and translation transformations, image data is transformed relative to the scale, orientation and location that has been assigned for each feature and all the future operations are performed on that transformed data.

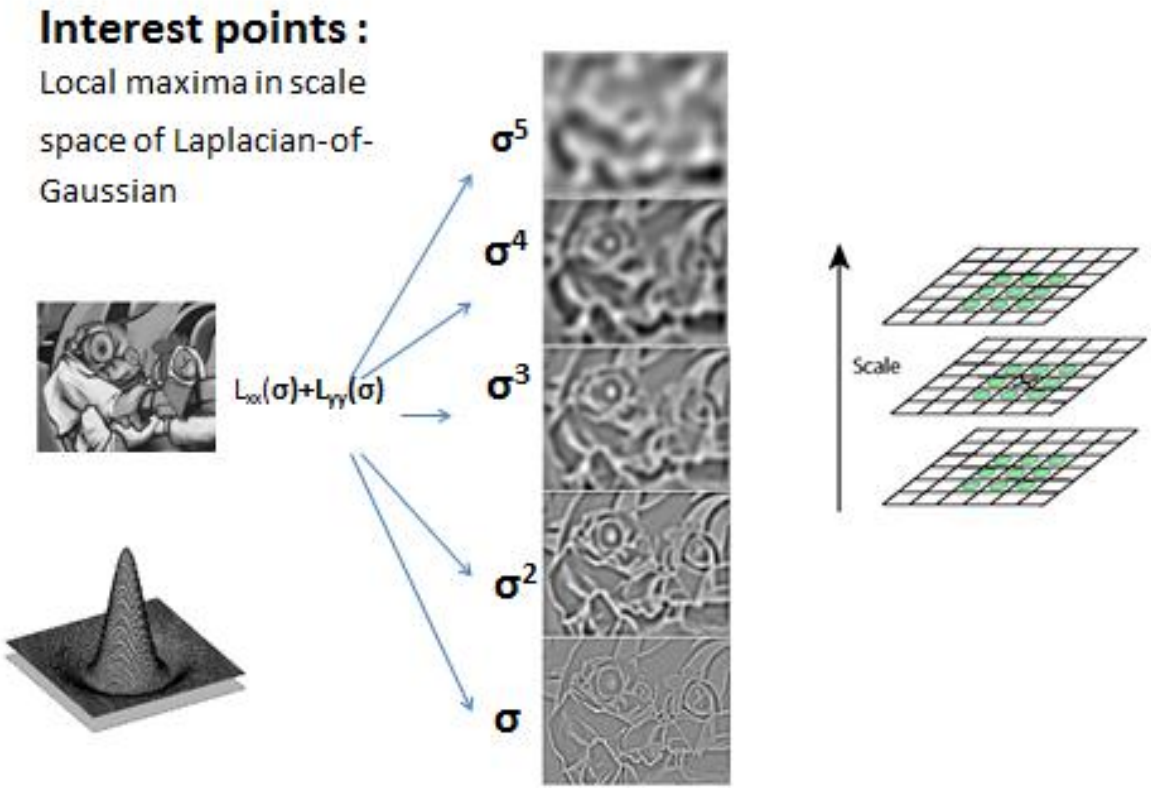
4. **Key point descriptor:** Descriptors are evaluated around each interest point in order to achieve invariance to illumination. For each interest point there is a vector of 128 values which we call a descriptor.

As this approach is Scale Invariant Feature Transform (SIFT), it transforms data of the image into scale-invariant coordinates relative to local features.

2.2 Detection of scale-space extrema

Create a scale space according to the paper scale-space filtering proposed by Adrew P. Witkin []. Apply whole spectrum of scales. Observe the zero crossings per scale in a scale space. For a given signal we can get the top level description from the scale space by removing the nodes which are stable. Iteratively remove the nodes which are less stable than any of their parents and off springs.

For two dimensional images, Laplacian of Gaussian will give the local maxima in scale-space and this local maxima is called as interest points of the image.



In building a scale-space, the important steps are examining all the scales to identify scale invariant feature and computing Laplacian Pyramid (Difference of Gaussian). For an image shown in the above figure, apply Gaussian filter, perform smoothing operation with different sigma values. Now, form Difference of Gaussian set by subtracting consequent scales. To decide the point as an interest point, consider a 3x3 neighborhood in the current scale, above, and below scales, compare the point with its 26-neighboring points. If the point is either local maxima or local minima, consider it as a potential SIFT interest point of the scale. By this analysis, we extract key points i.e., interest points from a two dimensional image.

Approximation of Laplacian of Gaussian by Difference of Gaussians,

$$Consider\ the\ heat\ equation:\ \frac{\partial G_S}{\partial \sigma} = \sigma \nabla 2GS$$

$$\sigma \nabla 2GS = \frac{\partial G_S}{\partial \sigma} \approx \frac{G_s(x, y, k\sigma) - G_s(x, y, \sigma)}{k\sigma - \sigma}$$

We can write,

$$GS(x, y, k\sigma) - GS(x, y, \sigma) = (k - 1)\, \sigma 2 \nabla 2 GS.$$

Typical values of $\sigma = 1.6$ and $k = \sqrt{2}$

Then create sequence of octaves by subsampling. Form difference of Gaussians by subtracting adjacent scales. And analyze for interest points as explained above.

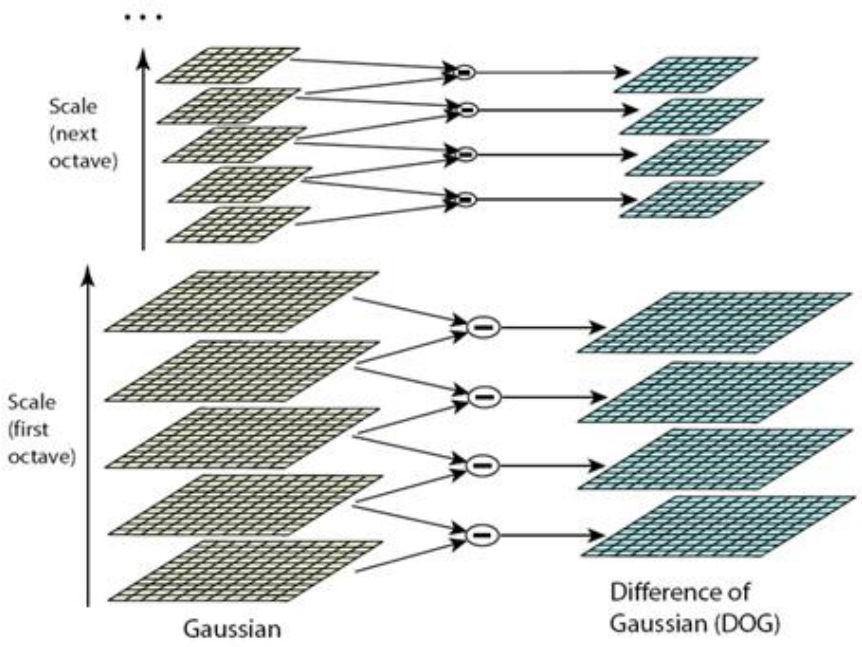


Figure 2.1 Scale space analyses

S(x, y, σ) is the function of scale-space of an image and it is calculated by the convolution of a Gaussian function with variable scale, G_S(x, y, σ), and input image, I(x, y):

$$S(x, y, \sigma) = [GS(x, y, \sigma) * I(x, y)],$$

$$Where, GS(x, y, \sigma) = \left(\frac{1}{2\pi\sigma 2}\right) exp\left(-\frac{x^2 + y^2}{2\sigma 2}\right).$$

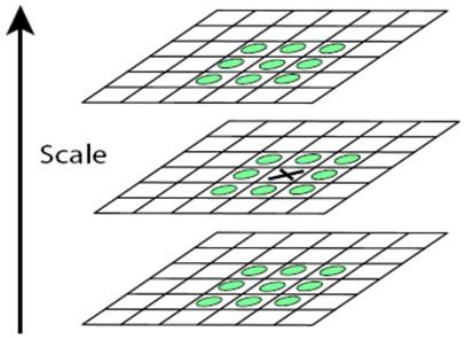


Figure 2.2 Selecting local minima or local maxima in 26 neighborhoods

In the above equation, the $(k-1)$ factor is defined as constant value different scales and therefore does not affect the extrema location.

2.2.1 Local extrema detection

To decide the point as an interest point, consider a 3x3 neighborhood in the current scale, above, and below scales, compare the point with its 26-neighboring points(shown in figure 2.2). If the point is either local maxima or local minima, consider it as a potential SIFT interest point of the scale. By this analysis, we extract key points i.e., interest points from a two dimensional image. Large number of extrema is computationally expensive. So, detect the most stable subset with a coarse sampling of scales.

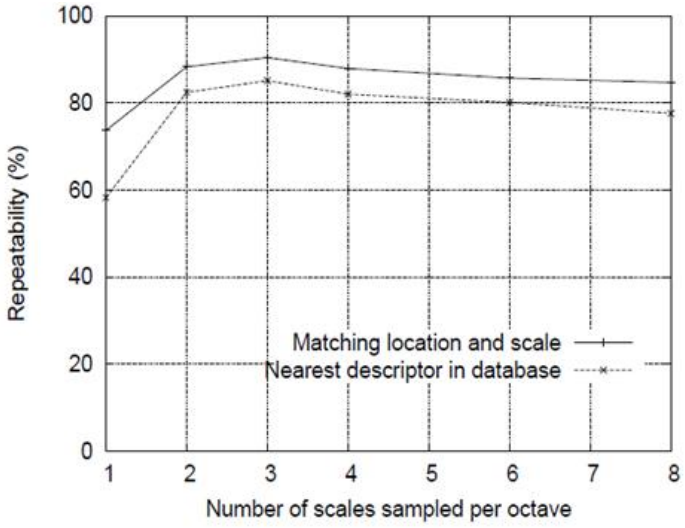


Figure 2.3 Graph between number of scale sampled per octave and percentage off repeatability

2.3 Accurate keypoint localization

Next is to localize each interest point. That is, deciding what is best scale and what is location x, y and σ. In the following figure 2.4, (a) shows the original image, (b) shows the arrows which represents local extrema. There can be some outliers due to low contrast, poorly localized candidates along the edge. In order to avoid those outliers, compute Taylor series expansion for DOG.

$$D(x) = D + \frac{\partial(D^T)}{\partial X} X + \frac{1}{2} X^T \frac{\partial^2 D}{\partial X^2} X \ldots\ldots\ldots (2)$$

Where, D and its derivatives are calculated at sample point and the offset from this point is given by $x = (x,y,\sigma)^T$. we have the function D(x), differentiate it and equate it to zero in order to get extrema location.

$$\hat{x} = -\frac{\partial^2 (D^{-1})}{\partial x^2} \frac{\partial D}{\partial x} \ldots\ldots\ldots\ldots\ldots\ldots\ldots (3)$$

The value of D(x) at extrema should be large, |D(x)| > threshold.

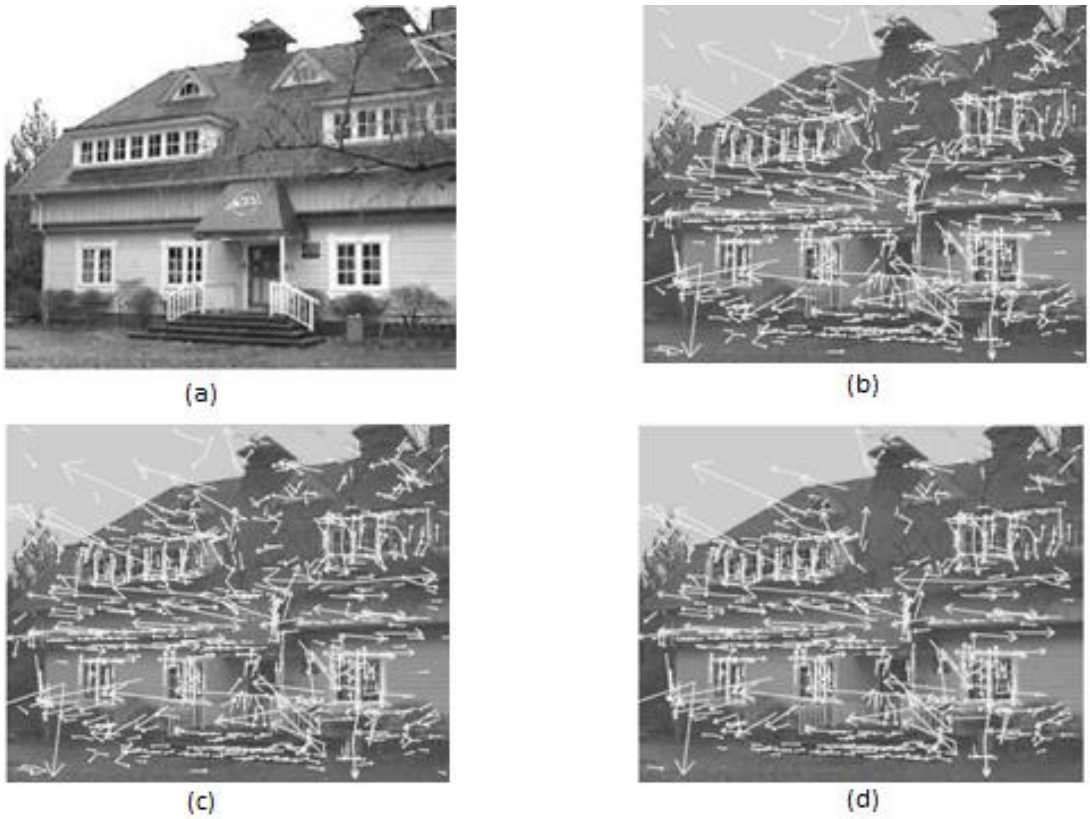


Figure 2.4 (a) original image. (b) 832 keypoints extracted in the image. (c) obtained 729 keypoints after applying a minimal contrast thresholding . (d) final 536 keypoints that remain after applying ratio of principal curvatures thresholding .

The approximation of Hessian and differentiation of function D is done by using subtraction of neighboring sample points. The approximation results in a 3x3 linear system. Wrong selection of interest point will occur when the x^ is greater than 0.5, because of mis-matching of sample point.

For rejecting unstable extrema with low contrast, the function value D(x^) at extrema is used. The rejection can be obtained as,

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

By considering the magnitude of $D(\hat{x})$ as threshold value, we discard the interest points which are less than the threshold.

The effect of keypoint or interest point selection on a natural image is shown in figure 2.4. A low-resolution 233x189 pixel image is used in order to avoid too much clutter and keypoints or interest point are indicated with arrows giving the position, direction and scale of each interest point. Figure 2.4 (a) shows the original image. The subsequent figures show the original image with reduced contrast. In figure 2.4 (b), 832 keypoints at all detected extrema of the DOG function is shown, while figure 2.4 (c) shows 729 interest points that remain after thresholding. The thresholding operation done in figure 2.4 (d) is explained in the following section.

2.4 Orientation assignment

In order to achieve invariance to image rotation, keypoint descriptor can be represented to the orientation, assigned to each keypoint on the basis of local image properties. The procedure is that the Gaussian smoothed image, L, is selected according to the scale of the key point with the closest scale, in order to perform all computations in scale-invariance manner. The gradient magnitude and orientation, m(x,y) and θ(x,y) respectively, for L(x, y), at this scale, are calculated using

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

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$

Around the key point region, a histogram of orientations is formed by considering gradient orientations. The histogram of orientations includes 36 bins covering 360 degree. Histogram is formed by adding gradient magnitude of corresponding gradient orientation of each sample.

Select the highest peak as the direction or orientation of the interest point. If there are multiple peaks in the histogram, the orientation of the key point is decided by considering the peaks that are 80% of the highest peak and taking resultant of them.

Figure 2.5 shows the stability of the interest point for location, scale, and orientation assignment under different noise conditions. The gap between the top two lines indicates that even after the addition of ±10% pixel noise the orientation assignment shows accuracy of 95% of the time. For correct matches, the variance of orientation is measured about 2.5 degrees and for 10% of noise it is 3.9 degrees. The bottom line shows the percentage of correct matching of an interest point or key point descriptor when matched to a 40,000 interest point or keypoints dataset. The graph shows that, even for large amounts of pixel noise, the SIFT features are resistant and the main cause of false detection is initial scale and location detection.

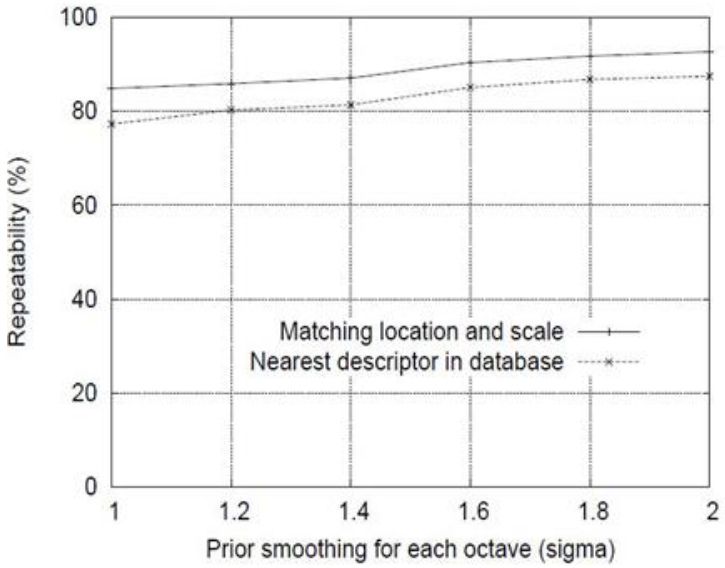


Figure 2.5 Graph drawn between prior smoothing for each octave and percentage of repeatability

2.5 The local image descriptor

Once we have interest points we have to calculate descriptor around it. There are many approaches available to calculate descriptors. One of the possible approaches is consider a window around the interest point or key point and extract the intensity values or the color values since they are sensitive to lighting changes, 3D object transformation. Use gradient orientation histograms as they provide robust representation. This approach is similar to human visual system. Adapting the features of human visual system, the approach experimentally shows good recognition accuracy for 3-dimensional objects rotated in depth by up to 20^0.

For extraction of local image descriptors at each interest point or key point, consider a 16x16 neighborhood around the interest point, calculate relative orientation and magnitude. Divide the neighborhood into 4x4 regions and compute weighted histogram. We will get a vector of 128 values consisting weighted histogram of 16 4x4 sub regions. The below example explains the procedure for 8x8 neighborhood with 2x2 sub regions. Take the 16 histograms each in 8 dimensions put them in a vector with numbers. Normalize the vector to make it to unit vector that will help to get illumination invariance. For non-linear intensity transforms, remove large gradients and renormalize to unit vector.

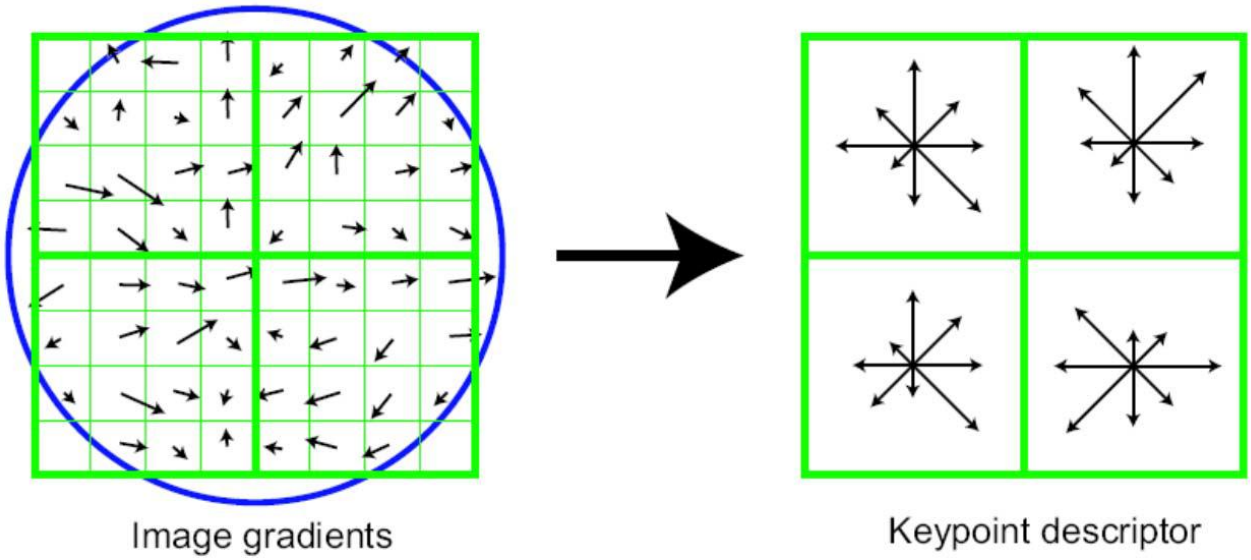


Figure 2.6 Image gradients and key point descriptor

2.5.1 Descriptor representation

The keypoint descriptor's computation is explained in figure 2.6.around the region of key point location, the image gradient magnitudes and orientations are sampled to select the level of Gaussian blur for the image, using the scale of the keypoint. To achieve orientation invariance, the descriptor coordinates and the gradient orientations are rotated relative to the orientation of key point. For efficiency, for all pyramid levels, the gradients are computed as described in Section 2.5. At each sample location, the gradients are shown with small arrows, in the Figure 2.6.

The Figure 2.6 shows the representation of weighted orientation for each sample point. This is done by using Gaussian window. The key point descriptor is shown. By creating the orientation histograms over 4x4 sample regions, it allows for a significant shift in gradient positions.

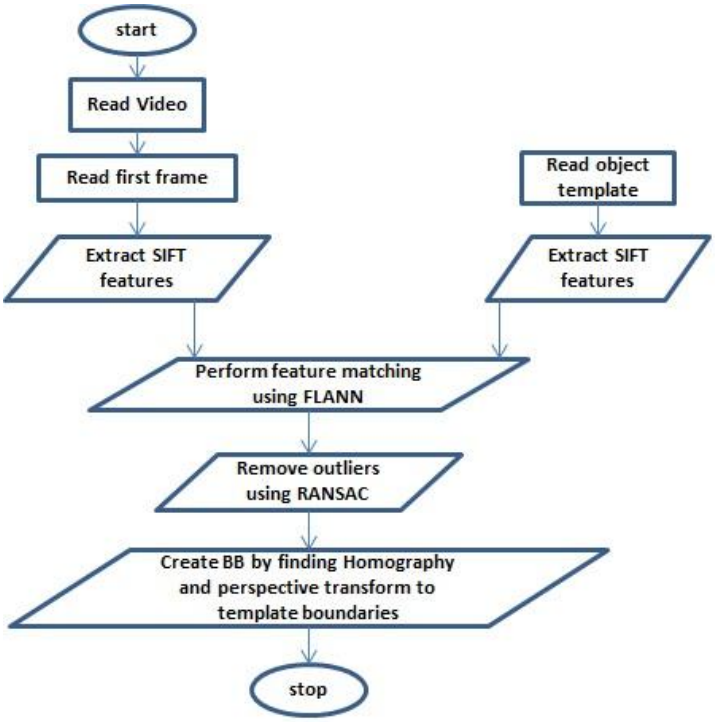
Finally, in order to reduce the effects of illumination change, the feature vector is modified. At first take the 16 histograms each in 8 dimensions put them in a vector with numbers. Normalize the vector to make it to unit vector that will help to get illumination invariance. For non-linear intensity transforms, remove large gradients and renormalize to unit vector.

2.6 Homography

The concept of homography relates the points on one plane to that on the other. This is given by a linear transformation using the homography matrix H. The homography matrix H transforms the homogenous coordinates of the points on one plane to that on the other. The points in 2D are given in their homogenous coordinates by appending a 1 at the end. Hence the location of a point given in 2D coordinates as $(x, y)^T$ is given in homogenous coordinates as $(x, y, 1)^T$. Let the a point be viewed from two standpoints of the camera. The locations of the points are x and the corresponding location of the same point in the other view is x'. The first point is located on a plane P_1 and the second on a plane P_2 . The homography matrix is now capable of relating the points in these two coordinates systems. It is an invertible matrix as described by Hartley and Zisserman [11] and is defined up to and arbitrary scale. It has eight degrees of freedom and is can be found out by having four point correspondences. These four point correspondences must be non-collinear.

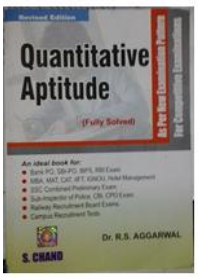
2.7 RANSAC

RANSAC is a model fitting algorithm capable of finding the data points that fit the model approximately, even in the presence of outlying data points. The RANSAC algorithm is an iterative algorithm that checks a hypothesis against the dataset to find how well the observed data fits the hypothesis. The main advantage of RANSAC is its ability to figure out the inlying data points even if the data is corrupted by noise of by high margin. It was first proposed by Fischlar and Bolles [10] and has been since been used extensively for the purpose of model fitting. It randomly samples the data to form a hypothesis and checks to see how many of the remaining data points concur with the assumed hypothesis. This is done repeatedly and that hypothesis that agrees the most with the data is our estimated model.



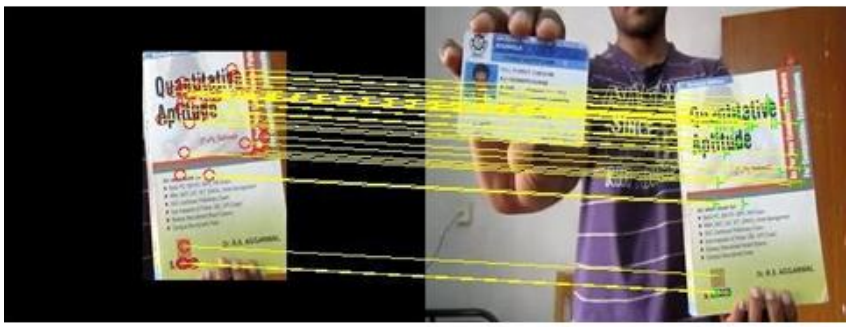
Flow chart for Recognition

Step wise experimental results for object recognition:



Template **Scene**

(a)



SIFT Feature Matching

(b)

Object Recognition

(c)

Figure 2.7 Object recognition experimental result

Chapter 3

Kanade Lucas Tomasi (KLT) Tracker

3.1 Introduction

Tracking means estimating the motion or trajectory of an object in the image plane as it moves around a scene.

Tracking can be categorized into:

- Object tracking (car, person, airplane, etc.)
- Feature tracking (Harris corners, SIFT, SURF)
- Single object tracking
- Multiple object tracking
- Tracking in fixed camera
- Tracking in moving camera
- Tracking in multiple cameras

Examples of tracking:

Figure 3.1 Tracking object contour

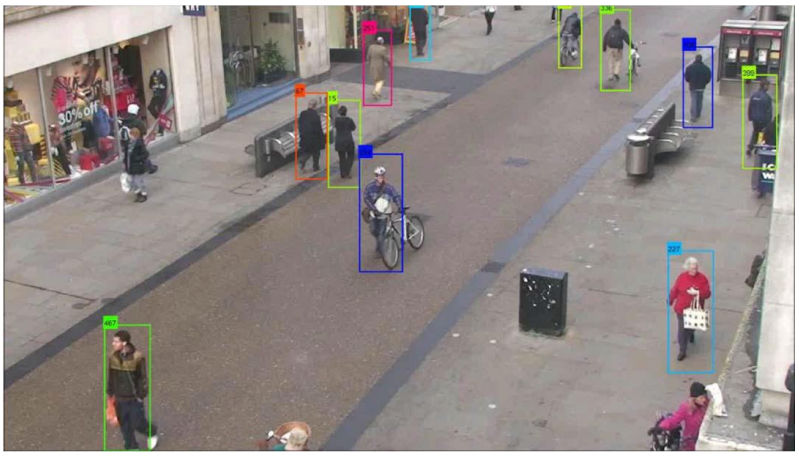


Figure 3.2 Person tracking with Bounding Box

Tracking is a vast topic, with extensive research being done currently. We have different tracking techniques like background subtraction based algorithms, feature based tracking algorithms. SIFT based tracking, CAM shift, Mean shift, KLT tracker are some feature based tracking techniques. KLT tracker algorithm works on the basis of Optical Flow.

3.2 Optical Flow

To analyze a video, the minimum number frames we require are two. One of the important basic operations in video analysis is Optical Flow. Optical Flow is a 2-dimensional vector that gives displacement of each pixel compared to previous frame. Optical Flow is proposed by Horn & Schunck.

3.2.1 Computing Optical Flow

***Horn & Schunck Optical Flow*:** For a video, we have 3-dimensional function *f(x,y,t)*, where (x,y) are coordinates of the frame and t is the frame number, in a frame. Assuming brightness constancy for two consequent frames, we can write the same point in the next frame as,

$$f(x,y,t) = f(x+dx, y+dy, t+dt)$$

By Taylor series expansion, we can write,

$$f(x,y,t) = f(x,y,t) + \frac{\partial f}{\partial x}dx + \frac{\partial f}{\partial y}dy + \frac{\partial f}{\partial t}dt$$

Neglecting higher order terms and rewriting the equation to,

$$f_x dx + f_y dy + f_t dt = 0$$

Substituting $\frac{dx}{dt}$ = u and $\frac{dy}{dt}$ = v,

$$f_x u + f_y v + f_t = 0$$

This is the equation of Optical Flow, solving for u and v gives us a

Lucas Kanade (Least Squares) method for solving Optical Flow equation:

Optical Flow equation,

$$f_x u + f_y v = - f_t$$

Since solving for two unknowns with one equation is a difficult task. We consider a 3X3 window around the point, so that we can write nine Optical Flow equations as,

$$f_{x1} u + f_{y1} v = - f_{t1}$$

$$f_{x2} u + f_{y2} v = - f_{t2}$$

.

.

$$f_{x9} u + f_{y9} v = - f_{t9}$$

Now we have nine equations for solving two unknowns. Rewriting the above equations in to matrix form,

$$\begin{bmatrix} fx1 & \cdots & fy1 \\ \vdots & \ddots & \vdots \\ fx9 & \cdots & fy9 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} ft1 \\ ft2 \end{bmatrix}$$

Rewriting the equation by some considerations,

$$Au = f_t$$

Multiply with transpose of A on both sides,

$$A^T Au = A^T f_t$$

$$u = (A^T A)^{-1} A^T f_t$$

With this equation we can find the solution for u and v.

Square minimum equation of Optical Flow,

$$\min \sum (f_{xi}u + f_{yi}v + f_{ti})^2$$

solving for u and v,

$$\sum (f_{xi}u + f_{yi}v + f_{ti})f_{xi} = 0$$

$$\sum (f_{xi}u + f_{yi}v + f_{ti})f_{yi} = 0$$

$$\sum f_{xi}^2 u + \sum f_{xi} f_{yi} v = -\sum f_{xi} f_{ti}$$

$$\sum f_{xi} f_{yi} u + \sum f_{yi}^2 v = -\sum f_{yi} f_{ti}$$

$$\begin{bmatrix} \sum fxi2 & \sum fxi\, fyi \\ \sum fxi\, fyi & \sum fyi2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum fxi\, fti \\ \sum fyi\, fti \end{bmatrix}$$

$$\begin{bmatrix} u \\ v \end{bmatrix} = -\begin{bmatrix} \sum f_{xi}^2 & \sum f_{xi} f_{yi} \\ \sum f_{xi} f_{yi} & \sum f_{yi}^2 \end{bmatrix}^{-1} \begin{bmatrix} \sum f_{xi} f_{ti} \\ \sum f_{yi} f_{ti} \end{bmatrix}$$

$$\begin{bmatrix} u \\ v \end{bmatrix} = \frac{1}{\sum f_{xi}^2 \sum f_{yi}^2 - (\sum f_{xi} f_{yi})^2} \begin{bmatrix} -\sum f_{yi}^2 & \sum f_{xi} f_{yi} \\ \sum f_{xi} f_{yi} & -\sum f_{xi}^2 \end{bmatrix} \begin{bmatrix} -\sum f_{xi} f_{ti} \\ -\sum f_{yi} f_{ti} \end{bmatrix}$$

Finally, the solution for Optical Flow is,

$$u = \frac{-\sum f_{yi}^2 + \sum f_{xi} f_{ti} + \sum f_{xi} f_{yi} \sum f_{yi} f_{ti}}{\sum f_{xi}^2 \sum f_{yi}^2 - (\sum f_{xi} f_{yi})^2}$$

$$v = \frac{-\sum f_{xi}^2 + \sum f_{y_i} f_{ti} + \sum f_{xi} f_{yi} \sum fxi\, fti}{\sum f_{xi}^2 \sum f_{yi}^2 - (\sum f_{xi} f_{yi})^2}$$

Lucas-Kanade Optical Flow method works only for small change in motion. The brightness changes rapidly if object moves faster. To compute Optical Flow vectors for large motions, image pyramids can be used. Applications of Optical Flow include motion based segmentation, structure from motion (3D shape and motion), alignment (global motion compensation) and video compression.

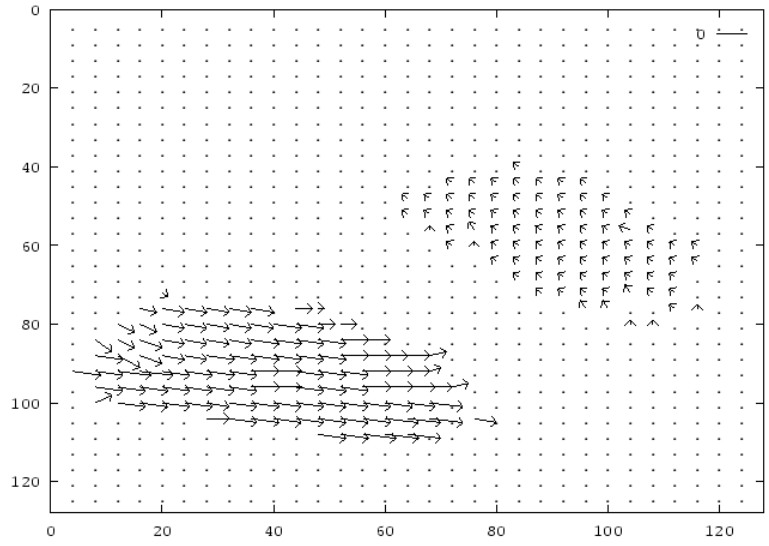
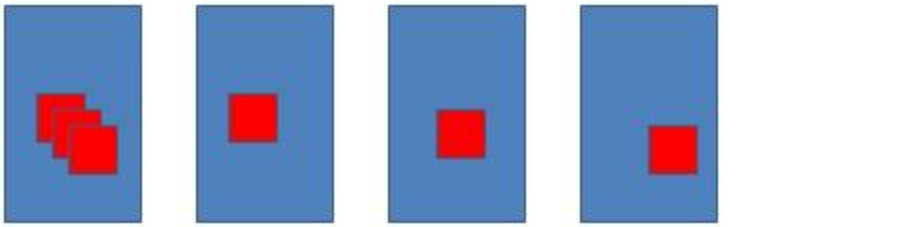
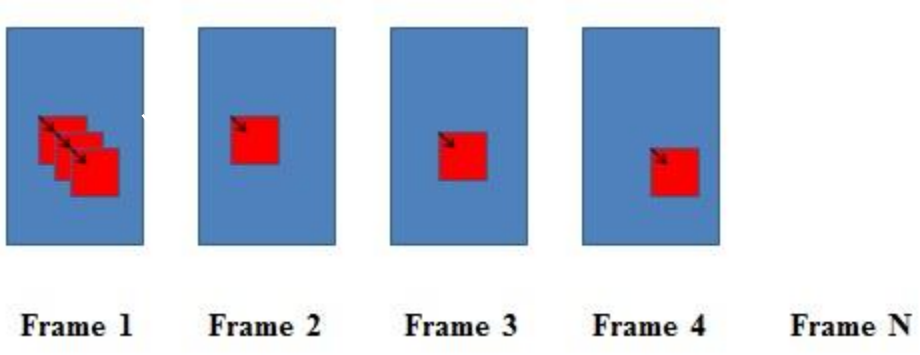


Figure 3.0.3 Optical Flow of two consequent frames

Now consider a video, in which an object is moving ass shown below,



Our task is to track the points on that object over the frames. If we find Optical flow for the object then we'll be able to track the object.

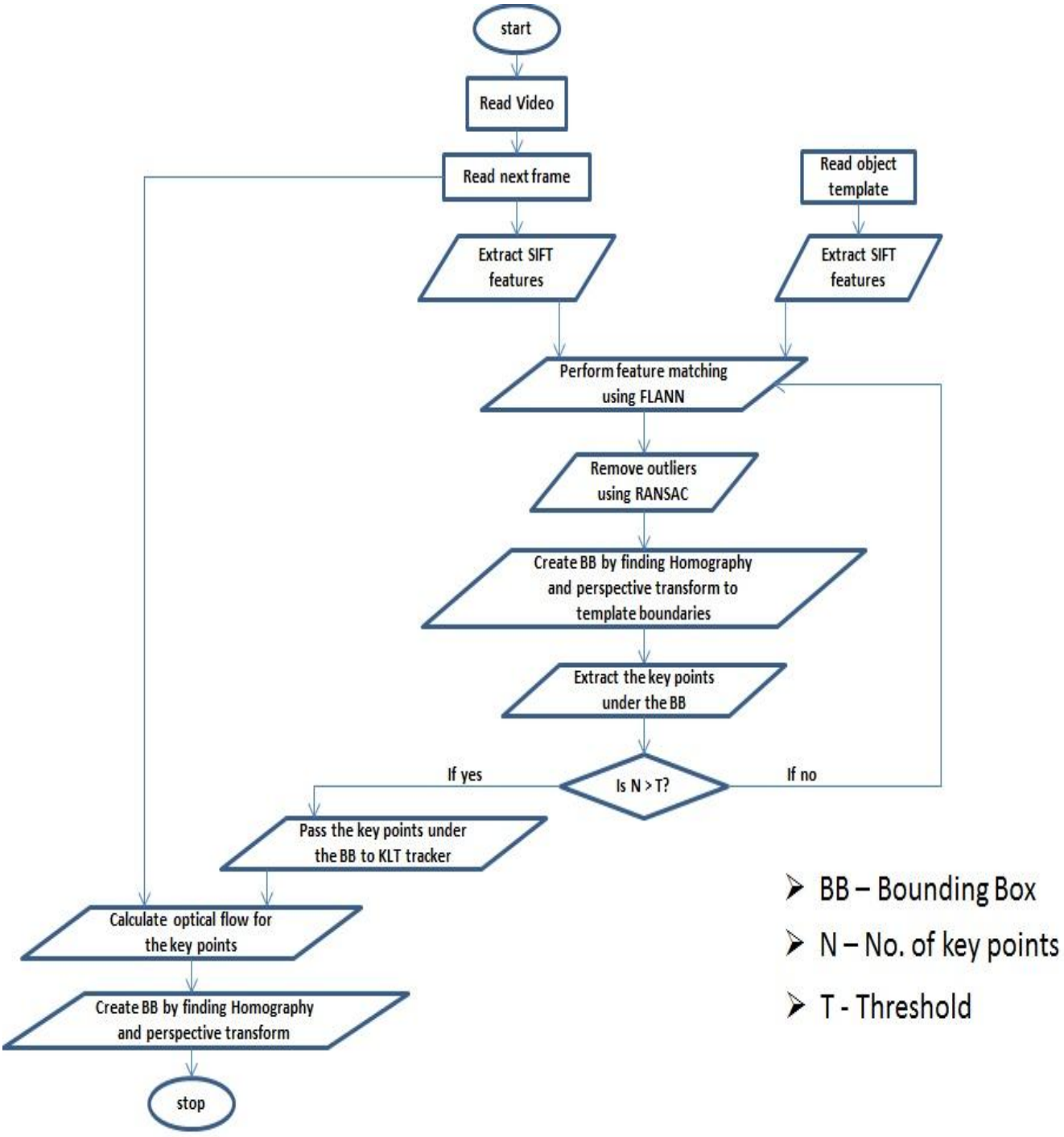


3.3 Simple KLT algorithm:

At the high level it is extremely simple. Detect Harris corners in the first frame. Then for each Harris corner we compute the motion i.e., local motion using optical flow for consecutive frames. Link these motion vectors from frame to frame to get the tracks. There is a chance that these corner points may disappear or other objects may appear. So, detect Harris points for every 10 or 15 frames.

- Detect Harris corners in the first frame
- For each Harris corner compute motion(affine or translation) between consecutive frames
- Link motion vectors in successive frames to get tracks
- Introduce new Harris points for every 10 or 15 frames

Flowchart for SIFT based object recognition and KLT tracker based tracking:



Flowchart for object recognition and tracking

Tracking results are shown in montage form from first to last (462) with an interval of 45 frames.

Figure 3.4 Object tracking result

Chapter 4

Implementation and results

4.1 SIFT Features

SIFT features are diacritic features that are not variant to affine transformations, illumination and 3D view point. There are four steps to compute SIFT features, (a) finding scale-space extrema, (b) localization of key point, (c) assigning orientation, (d) creating descriptor. They are complex to compute but effective and robust to occlusion, clutter or noise.

4.2 Modified KLT Tracker

KLT tracker is point tracker which detects Harris corners in the frame and computes Optical flow (motion vectors) for each corner point in the next frame. To get tracks link motion vectors in successive frames. In this way, kanade-Lucas-Tomasi proposed a simple yet efficient tracking system.

Here, instead of detecting Harris corners we use SIFT keypoints under the bounding box in the first frame. For each keypoint we compute motion (translation or affine) between consecutive frames. By linking motion vectors in successive frames we get tracks. When number of keypoints becomes less than threshold, we performed feature matching between current frame and target image and proceeded as mentioned above.

4.3 Proposed Method

For hardware implementation of object recognition and tracking, we modified the KLT tracker algorithm.

The procedure is given below:

i. SIFT Feature Extraction

ii. Feature Matching

iii. Calculation of Homography [11] and Perspective transform

iv. Creating bounding box

v. Calculating Optical Flow [5] for the points under bounding box

The step wise execution of the algorithm is shown in Fig.2.

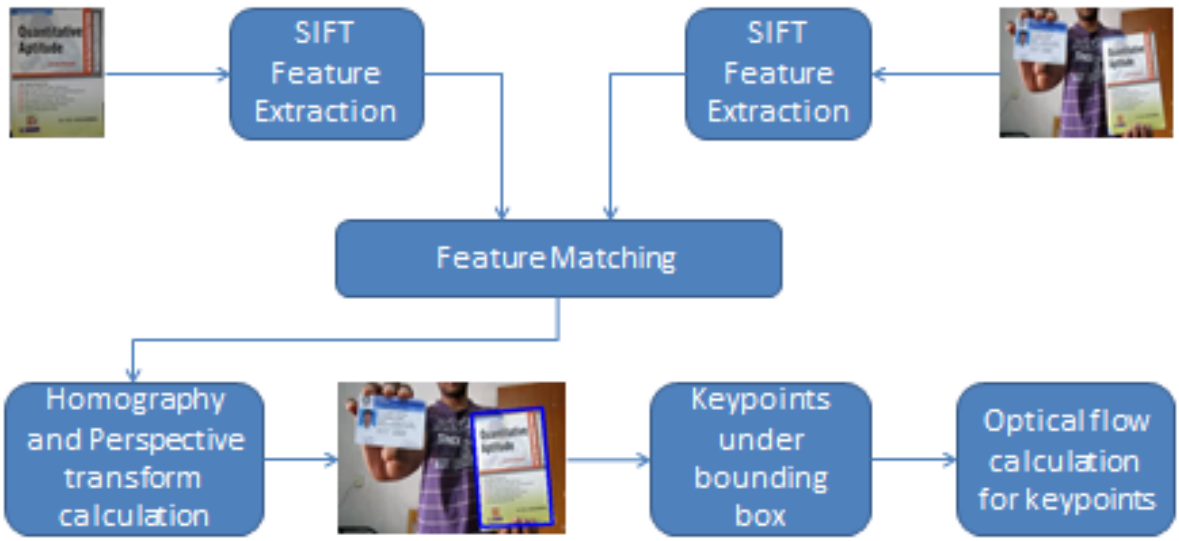
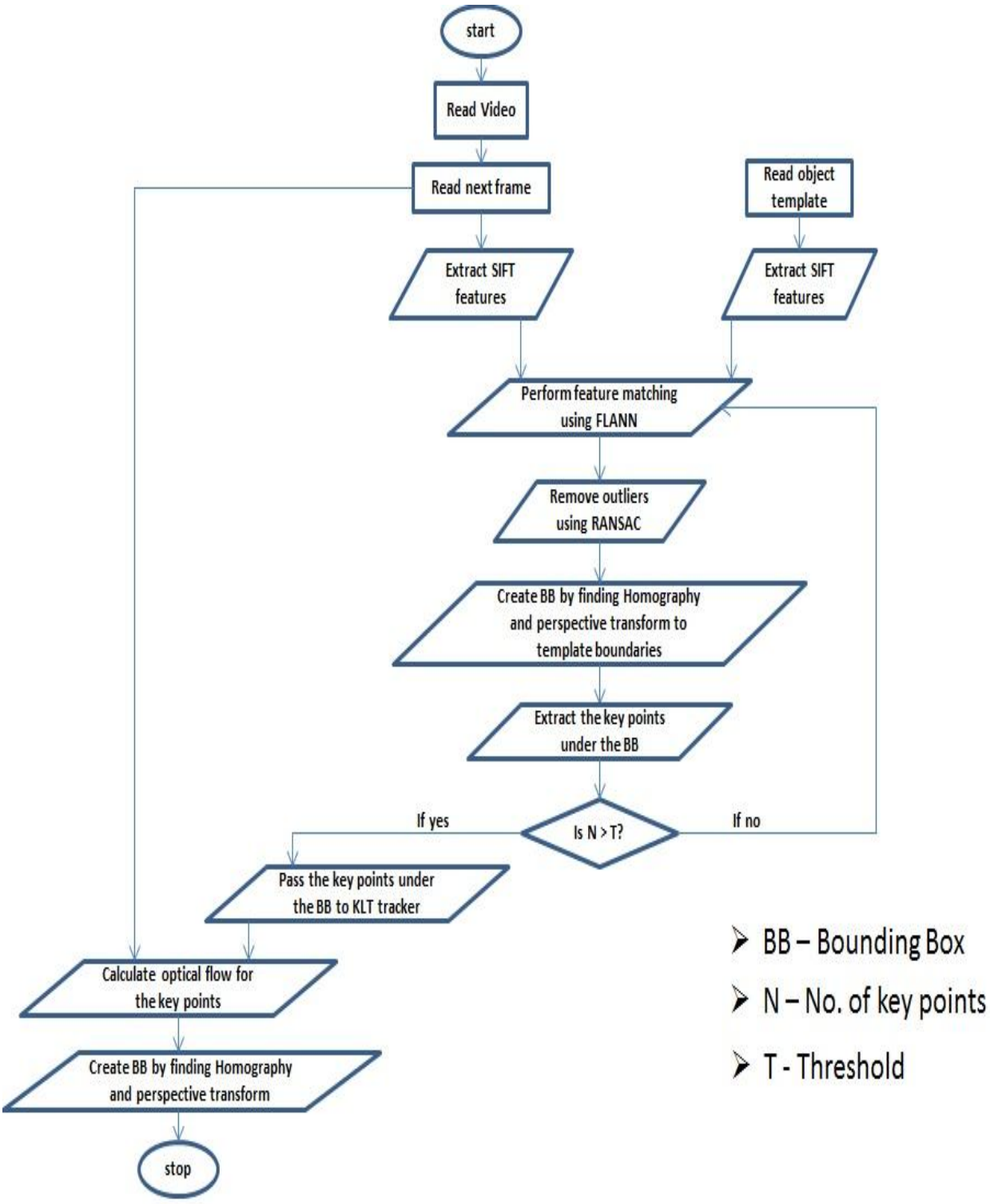


Figure 4.1 Proposed algorithm

Flow chart for object recognition and tracking:



4.4 Experimental Results

Figure 4.2 Recognition and tracking of object 1 in indoor environment

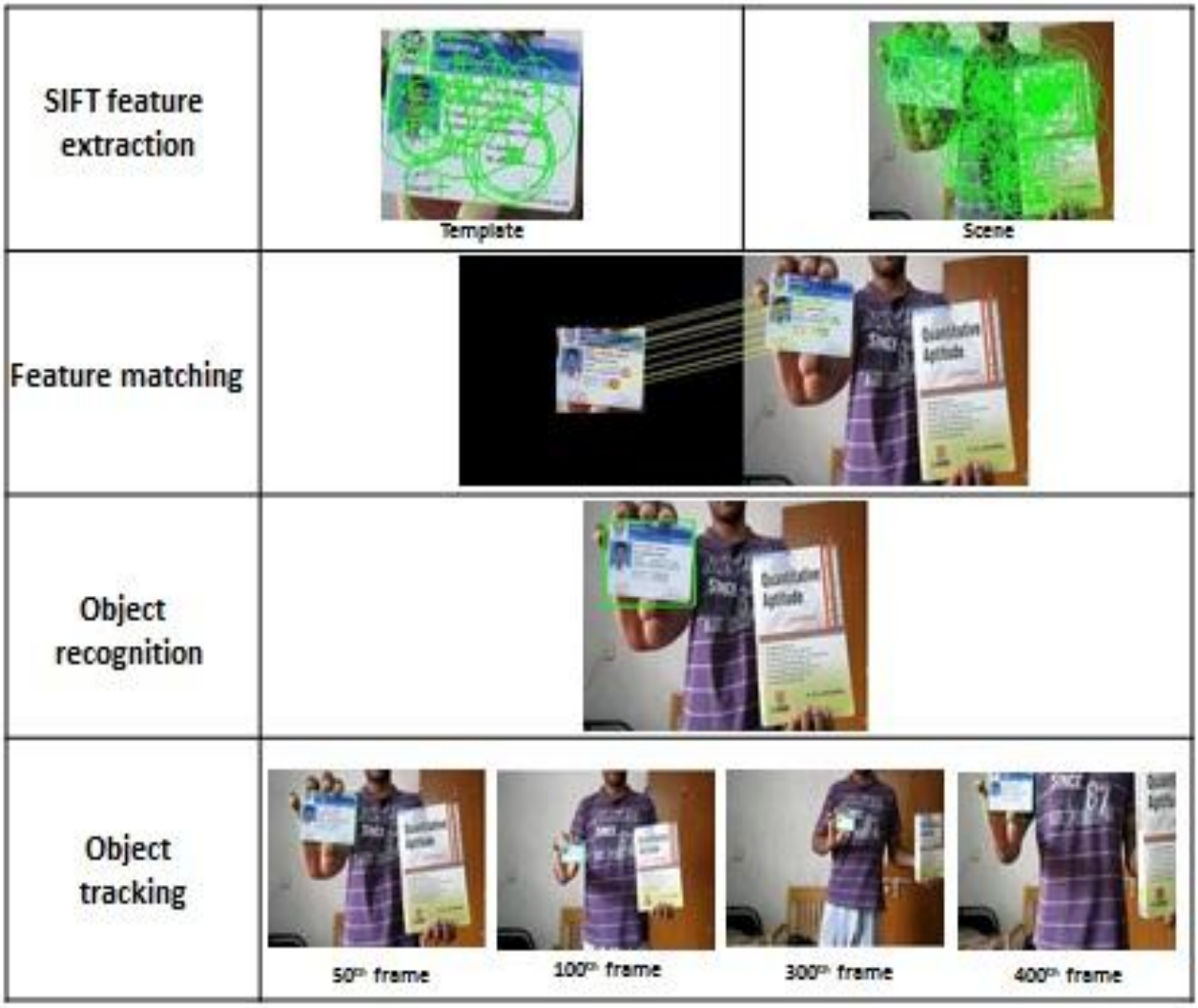


Figure 4.3 Recognition and tracking of object 2 in indoor environment

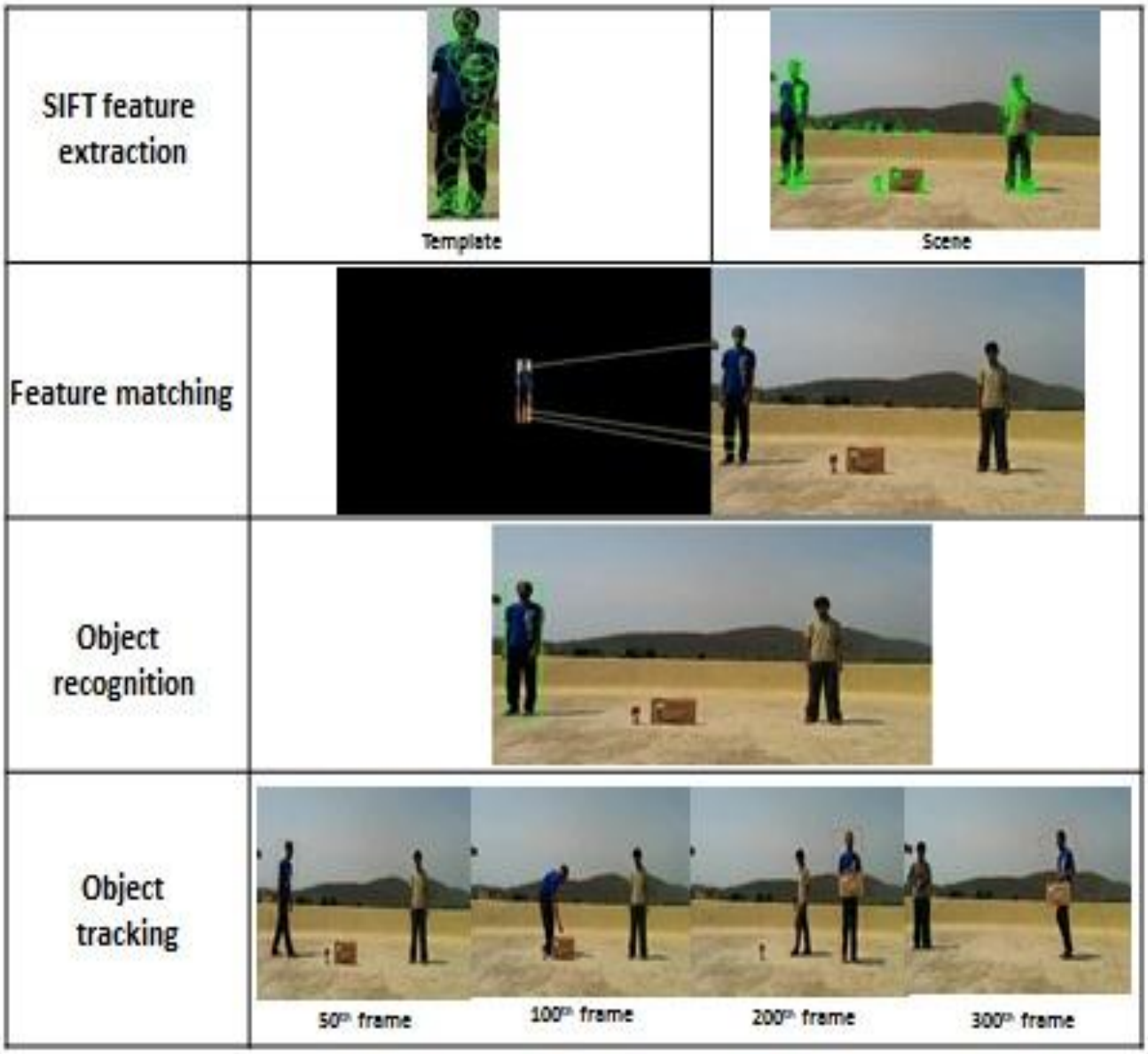


Figure 4.4 Object recognition and tracking in outdoor environment

4.5 System Design

4.5.1 Hardware description:

The system consists of a USB camera and Arduino board connected to PC. USB camera is mounted on an assembled two servo motors. The Arduino board runs at 16 MHz and sends PWM signals to control servo motors.

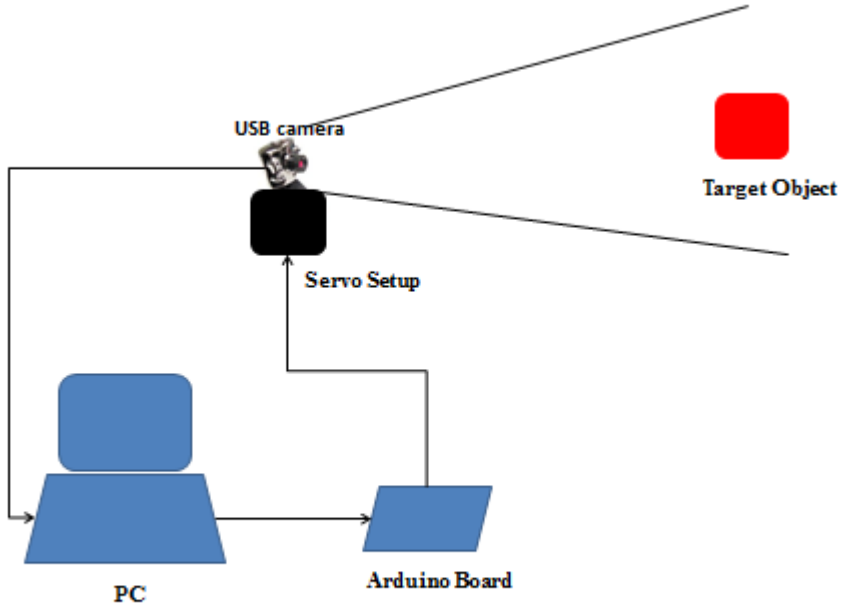


Figure 4.5 System design overview

USB camera – records motion of target object

PC – performs Image processing

Arduino Board – sends PWM control signals to servos

Servo setup – moves camera based on control signals

4.5.2 Software system design

Our implementation uses openCV and Python libraries to perform image processing operations in PC and Arduino sends the appropriate PWM signals to servo motors. A Python code is executed in PC and as per the result Arduino sends controls signals to servos.

4.6 Hardware Implementation

Arduino is an ATMEGA 328 micro-controller works at a frequency of 16MHz. For the required task, the setup consists of PC, USB camera mounted on two servo motors, Arduino board. USB camera records motion of target object. Based on the data from USB camera, we performed real-time object recognition and tracking in PC. According to the image processing results, we send control signals to servo setup through Arduino board. Servo setup makes USB camera to follow the target object based on control signals. Hardware setup is shown in Fig.3.

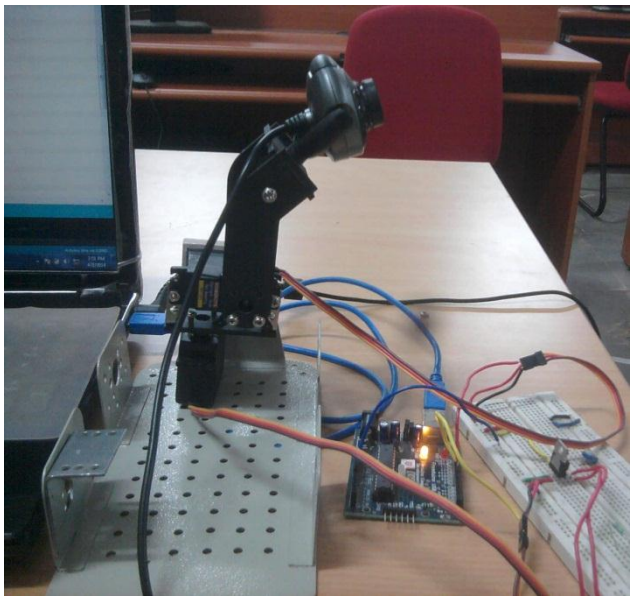


Figure 4.6 Hardware setup

4.6.1 Procedure for camera motion

Make the servo motors (pan,tilt) fix at (90^0,90^0) angles. Fix camera center as (x,y). Calculate target object's centroid as (x',y'). Compare (x,y) and (x',y')

i. If x'>x and y'>y, then decrease pan angle by step size, θ and tilt angle by step size, θ
ii. If x'<x and y'>y, then decrease pan angle by step size, θ' and increase tilt angle by step size, θ'
iii. If x'<x and y'<y, then increase pan angle by step size, θ' and tilt angle by step size, θ'
iv. If x'>x and y'<y, then increase pan angle by step size, θ' and decrease tilt angle by step size, θ'

Perform the increment or decrement operation until the camera center and bounding box's centroid matches.

4.6.2 Hardware experimental results

A USB camera and Arduino board are connected to PC. Two servo motors assembled such that one works for panning and other for tilting are connected to Arduino board. Servo motors are mounted with USB camera. The image processing tasks are performed in PC and the results are passed to Arduino board. Based on the results Arduino board sends corresponding control signals to servo motors.

Frames shown in Fig.4 (c) are from frame number 1 to frame number 270 at an interval of 15 frames. If we observe the results, from 1st frame to 8th frame, the camera view gets changed from left to right and from 9th to 16th frame; the camera view gets changed from bottom to top.

By this change of view, we can infer the camera has moved according to the target object in order to keep the centroid of the object at its center.

Figure 4.7 Real-time object recognition and tracking results

4.7 Limitations:

a) The target object should present in the frame when the camera starts image acquisition.

b) Start the execution only after setting the servo motors in their default position.

c) The camera stops following the target once the tracking fails.

d) The homography is calculated corresponding to the previous frame. So the bounding box shape gets deform.

e) The size of the target should be large enough to detect unless the recognition fails.

f) There is chance of mis-matching because of occlusion.

Chapter 5

Conclusion

5.1 Conclusion

In this thesis, we have proposed a real-time moving target following camera system, based on object recognition and tracking. The combination of SIFT and modified KLT tracker approach adopted here shows efficient recognition rate and is a computationally simple tracking system whose results are validated by the experiments. The tracking is robust to occlusions and affine transformations.

5.2 Future Scope

This project can be extended further to make a robot that can follow the target object. By using an embedded system like a Beagle bone board, we can perform the image processing tasks without the help of PC.

REFERENCE

[1] D. G. Lowe. “Distinctive image features from scale-invariant keypoints”. International Journal of Computer Vision, 60(2) :91–110, 2004.

[2] C. Tomasi and T. Kanade. “Detection and Tracking of Point Features”, Carnegie Mellon University Technical Report CMU-CS-91-132, 1991.

[3] M. Brown and D. Lowe.: Invariant features from interest point groups”. In British Machine Vision Conference, pages 656–665, 2002.

[4] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: “Speeded Up Robust Features. In ECCV (1), pages 404–417, 2006”.

[5] “Pyramidal Implementation of the Lucas Kanade Feature Tracker Description of the algorithm”, by Jean-Yves Bouget, Intel Corporation.

[6] Jianbo Shi and Carlo Tomasi, “Good features to track”, Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn., pages 593{600, 1994.

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

[8] C. Schmid, R. Mohr, and C. Bauckhage. “Evaluation of interest point detectors”. International Journal of Computer Vision, 37(2):151–172, June 2000.

[9] “SIFT Feature for Object Recognition and Tracking within the IVSEE System”, by Fernando Lopez-Garcia, IEEE, 2008.

[10] M.A. Fischler and R.C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM, 24(6):381–395, 1981.

[11] R. Hartley and A. Zisserman. Multiple View Geomerty in Computer Vision. Cambridge University Press, second edition, 2003.

[12] R. A. Brooks, "Model-based 3-D interpretation of 2-D images," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 5, pp. 140–150, May 1983.

[13] P. Fua and A. J. Hanson, "Using generic geometric models for intelligent shape extraction," in *Proc. DARPA Image Understanding Workshop*, Los Angeles, CA, 1987, pp. 227–233.

[14] H. G. Barrow and J. M. Tenenbaum, "MSYS: A system for reasoning about scenes," *Tech. Note 121, Artificial Intell. Cent., SRI Int*, Apr. 1976.

[15] T. M. Strat and M. A. Fischler, "Content-based vision: Recognizing objects using information from both 2-D and 3-D imagery," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 13, pp. 1050–1065, Oct. 1991.

[16] S. Ulman, *High-Level Vision-Object Recognition and Visual Cognition*. Cambridge, MA: MIT Press, 1996.

[17] D. B. Gennery, "Visual tracking of known 3-D objects," *Int. J. Comput. Vision*, vol. 7, no. 3, pp. 243–270, 1992.

[18] D. G. Lowe, "Robust model-based motion tracking through the integration of searching and estimation," *Int. J. Comput. Vision*, vol. 8, no. 2, pp. 113–122, 1992.

[19] F. Meyer and P. Bouthemy, "Region-based tracking using affine motion models in long image sequences," *Comput. Vision, Graphics, Image Proc.: Image Understanding*, vol. 60, no. 2, pp. 119–140, 1994.

[20] D. Koller, K. Daniilidis, and H. Nagel, "Model-based object tracking in monocular image sequences of road traffic scenes," *Int. J. Comput. Vision*, vol. 10, pp. 257–281, 1993.

[21] J. Malik, D. Koller, and J. Weber, "Robust multiple car tracking with occlusion reasoning," *Eur. Conf. Comput. Vision*, Stockolm, Sweden, 1994, pp. 189–196.

[22] G. L. Foresti, "A real-time system for video surveillance of unattended outdoor environments," *IEEE Trans. Circuits Syst. Video Tech.*, vol. 8, pp. 697–704, 1998.

[23] P. Maragos, "Pattern spectrum and multiscale shape representation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 11, pp. 701–716, July 1989.

[24] "Mean Shift: A robust approach towards feature space analysis", by Dorin Comaniciu and Peter Meer, IEEE transactions on pattern analysis and machine intelligence, 2002.

[25] "Incremental learning for robust visual tracking", by David A. Ross, Jongwoo Lim, Ruei-Sung Lin.

[26] "Tracking by sampling trackers" by Junseok Kwon and Kyoung Mu Lee, ICCV 2011 International conference.

[27] "Robust tracking using foreground background tracker" by Nguyen and Smeulders, International Journal of Computer Vision 69(3), 277–293, 2006.

[28] "Hough based tracking of non-rigid objects" by Godec, Roth, Bischof.

[29] "Tracking-Learning-Detection" by Zdenek Kalal, Mikolajczyk and Matas, IEEE transactions on pattern analysis and machine intelligence, vol. 6, no. 1, January 2010.

[30] "Structured output tracking with kernels" by Hare, Saffari, Torr, ICCV, 2011 International Conference.

[31] "Lucas-Kanade 20 Years on: A unifying framework" by Simon Baker and Matthews, International Journal of Computer Vision 56(3), 221–255, 2004.