

154614

İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

**MOTION DETECTION AND ROUTE MAPPING
IN CLOSED ENVIRONMENTS**

M.Sc. Thesis by

Ferhat UÇAN

504021505

Date of submission : 26 April 2004

Date of defence examination: 20 May 2004

Supervisor (Chairman): Assoc. Prof. Dr. Coşkun SÖNMEZ

Members of the Examining Committee Prof.Dr. Bülent ÖRENCİK (İTÜ.)

Prof.Dr. Tamer ÖLMEZ (İTÜ.)

MAY 2004

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

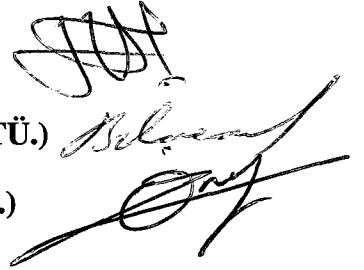
**KAPALI MEKANLARDA
HAREKET ALGILAMA VE ROTA ÇİZİMİ**

YÜKSEK LİSANS TEZİ
Ferhat UÇAN
504021505

Tezin Enstitüye Verildiği Tarih : 26 Nisan 2004
Tezin Savunulduğu Tarih : 20 Mayıs 2004

Tez Danışmanı :
Diğer Jüri Üyeleri

Doç.Dr. Coşkun SÖNMEZ
Prof.Dr. Bülent ÖRENCİK (İTÜ.)
Prof.Dr. Tamer ÖLMEZ (İTÜ.)



MAY 2004

PREFACE

I would like to record my appreciation to Assoc. Prof. Dr Coşkun SÖNMEZ for his helpful guidance with great importance. I wish to record my deep thanks to him for taking the time to provide significant comments throughout the implementation of this thesis. I would also thank to Barış ERGÜN, General Manager of Balya Limited for his supports. Special appreciation is due to all our teachers who are our guide for engineering education. Finally I would like to dedicate my thesis to my parents Hilal and Salih UÇAN and my sister Ayşe UÇAN. I owe much to my family for all their self-sacrifice, patience and support during all my education.

May 2004

Ferhat UÇAN

TABLE OF CONTENTS

ABBREVIATIONS	v
LIST OF FIGURES	vi
ÖZET	vii
SUMMARY	ix
1. INTRODUCTION	1
2. COLOR DEFINITION	4
2.1. Classification Of Colors	5
3. DIGITAL IMAGE PROCESSING	9
3.1. Moving Object Recognition Techniques	10
3.1.1. Background Difference Technique	10
3.1.2. Corresponding Pixel Comparing Technique	11
3.2. Image Cleaning Techniques	12
3.2.1. Convolution	15
3.2.2. Neighbourhood Averaging	15
3.2.3. Median Filter	17
3.3. Tresholding	18
4. IMPLEMENTATION OF THESIS	21
4.1. Getting Frames	22
4.2. Comparing Frames With Background	23
4.3. Cleaning Noises On Frames	25
4.4. Thresholding Frames	26
4.5. Calculating Mass Center	27
4.6. Geometric Calculation	30
4.7. Marking The Route	33
4.8. Interface Design	33
5. EXPERIMENT STATISTICS	40
6. SEGMENTATION USING BACKGROUND REGISTRATION	44
6.1. The Algorithm	45
7. HUMAN MOTION ANALYSIS	48
7.1. Application Fields	48
7.2. Previous Work	50
8. CONCLUSIONS AND RECOMMENDATIONS	55
REFERENCES	58
AUTOBIOGRAPHY	59

ABBREVIATIONS

DIP	: Digital Image Processing
RGB	: Red Green Blue
CYM	: Cyan Yellow Magenta
HSI	: Hue Saturation and Intensity
BMP	: Bitmap File Format
MSB	: Most Significant Bit
LSB	: Least Significant Bit
GUI	: Graphical User Interface
OOP	: Object Oriented Programming



LIST OF FIGURES

	<u>Page No</u>
Figure 1.1 System definiton.....	2
Figure 2.1 Wavelength diagram.....	4
Figure 2.2 RGB diagram.....	7
Figure 2.3 HSI diagram.....	8
Figure 3.1 Backround difference.....	10
Figure 3.2 Seperation the sub objects.....	11
Figure 3.3 64 grayscale histogram.....	12
Figure 3.4 Contrast stretching, Normalization.....	14
Figure 3.5 Adjusting the image histogram to improve image contrast.....	14
Figure 3.6 Mean Filter kernels.....	16
Figure 3.7 Low Pass Filter kernels.....	17
Figure 3.8 High Pass Filter kernels.....	17
Figure 3.9 Application of the median filter.....	18
Figure 3.10 Tresholding.....	20
Figure 4.1 Algorithm process.....	21
Figure 4.2 Subtraction image.....	24
Figure 4.3 Subtraction image.....	24
Figure 4.4 Mean filter application.....	25
Figure 4.5 Median filter application.....	26
Figure 4.6 Tresholding.....	27
Figure 4.7 A wrong mass center calculation.....	28
Figure 4.8 Calculated mass centers.....	29
Figure 4.9 Mass center on frame without a moving object.....	29
Figure 4.10 Measured object.....	30
Figure 4.11 Measuring operation.....	31
Figure 4.12 Geometric projection.....	32
Figure 4.13 Interface window.....	34
Figure 4.14 Interface window while process is running.....	34
Figure 4.15 Interface window with processed images.....	35
Figure 4.16 ‘File’ menu.....	36
Figure 4.17 ‘Edit’ menu.....	36
Figure 4.18 ‘View’ menu.....	36
Figure 4.19 ‘Settings’ menu.....	37
Figure 4.20 ‘Help’ menu.....	37
Figure 4.21 ‘Picture Settings’ window.....	38
Figure 4.22 ‘Analysis Settings’ window.....	38
Figure 4.23 ‘Flow Control’ window.....	39
Figure 4.24 ‘Help Menu’ window.....	39
Figure 4.25 Status Bar.....	39
Figure 4.26 Tool Bar.....	39
Figure 5.1 Test results.....	42
Figure 5.2 Percentage of success.....	43
Figure 6.1 Segmentation using background registration technique.....	45

KAPALI MEKANLARDA HAREKET ALGILAMA VE ROTA ÇİZİMİ

ÖZET

Güvenliğin önemi gittikçe artmaktadır. İnsanların ve diğer nesnelerin hareketinin takibi güvenlik sistemlerinde ayrı bir öneme sahiptir. Bu tip sistemlerde görüntü işleme teknikleri oldukça uygundur. Kameralarla çekilen görüntülerdeki hareketli cisimlerin kapalı mekandaki pozisyonlarının belirlenmesi ve rotasının çıkarılması kapalı mekanlardaki güvenlik için oldukça önemli ve faydalıdır. Bu tezin amacı da odadaki hareket eden cisimler için bir izleme mekanizmasının kurulmasını sağlamaktır. Sistem hareket eden cismi algılayabilmekte, oda içerisindeki pozisyonunu ve rotasını belirleyebilmektedir. Hareketli nesnenin pozisyonunu hatasız olarak belirleyebilmek için uygulamada iki kamera kullanılmıştır. Her kamera odanın iki boyutlu bir resmini görmektedir, fakat rotanın oda haritası üzerinde çıkarılabilmesi için üçüncü boyut olan derinlik bilgisine ihtiyaç duyulmaktadır. Derinlik boyutu ikinci bir kameranın kullanılmasıyla sağlanmaktadır.

Uygulamanın gerçekleştirilmesinde iki ana kısım söz konusudur:

İki kameradan alınan görüntülerin örneklenmesi: Bunun için istenen frekansta görüntü dosyalarını resim dosyalarına örnekleyebilen sinyal örnekleme uygulaması kullanılmıştır.

Görüntü İşleme: Bu adımda nesnenin tanınması ve iki kameraya olan açı ve uzaklıklarının bulunması işlemleri gerçekleştirilir. Bu adım dört alt adımdan oluşmaktadır:

- 1) Nesnenin odanın arka planından izole edilmesi. Bunun için resim odanın arka planıyla karşılaştırılarak fark alma işlemi yapılır. Arka plandan farklı pikseller hareket eden nesneleri ortaya çıkarır.
- 2) Değişik cisimler arasında seçim yaparak gerçek nesneyi belirleme. Önceki adımda elde edilen fark resmi taranır, piksel grupları nesnelere sınıflandırılır. En büyük nesne oda içerisindeki hareket eden cismi belirler.
- 3) Kameraların oda içerisindeki biçimi ve açısı yardımıyla nesnenin konumu en yüksek hassasiyette hesaplanır. Kameranın çektiği orjinal resimle karşılaştırılır ve kameranın saptırma parametreleri belirlenir, bu parametrelerden yararlanarak nesnenin oda içerisindeki pozisyonu belirlenebilir.
- 4) Nesnenin oda içerisindeki konumuna olan açısı hesaplanır. Bunun için bilinen uzaklıkta bir resim yardımıyla her bir dereceye kaç pikselin karşılık geldiği hesaplanır. Bu katsayı kullanılarak nesnenin konumu, açı ve yön bilgisine dönüştürülebilir.

Kapalı mekandaki şartların ve kameranın pozisyonunun değişmesi ihtimaline karşın ihtiyaçların ölçeklenebilir ve uyumlu ihtiyaçlar belirlenmiştir. Bu yüzden sistem modüler olarak nesneye dayalı yöntemlerle geliştirilmiştir. Böylece gelecekteki yeni

ihtiyalara ve deėiřikliklere kolay adapte olabilir bir sistem tasarlanmıřtır. Kamera konumu ve parametrelerindeki deėiřimleri g z n ne alarak kullanıcının g r nt  iřleme algoritmalarında etkili olacak bir takım parametreleri tasarlanan grafik aray z  ile deėiřtirmesine oplanak saėlanmıřtır. B ylelikle en uygun kamera parametreleri belirlenebilecektir.

Bu tezle g r nt  iřleme tekniklerini kullanarak alternatif bir g venlik sistemi oluřturulmuřtur. Bu modelde nesnenin hareketi takip edilmekte ve rotası ıkarılmaktadır. Tezde kullanılan g r nt  iřleme algoritmaları hareket eden nesneyi resimden ayırmayı saėlamaktadır. G r nt  iyileřtirme g r nt deki detayların ve g r n m n kuvvetlendirilmesi, g r nt  b l mlere, g r nt deki nesnelerin sınıflandırılması gibi alt iřlemleri iermektedir.



MOTION DETECTION AND ROUTE MAPPING IN CLOSED ENVIRONMENTS

SUMMARY

Security's importance is growing up each day. Tracking the movement of people and other objects is important in security system concept. Image processing methods are suitable for these systems. It will be useful for security of a covered area to locate the moving object on frames which captured by cameras and mark its route on the area's map. The goal of this thesis is designing and implementing a tracking system for moving object in a room. The system recognizes an object and marks its location and route on the room's map. In order to determine the absolute position of moving object, two cameras will be used. Each camera "sees" only a two-dimension picture while the map shown by the application represents the third dimension – the depth dimension. Third dimension can be gotten by using second camera.

The implementation is executed in two main phases :

Sampling pictures from two video cameras: Using an application for video signal sampling which enable to sample picture in the desirable moment into the clipboard.

Image processing: Object recognition and finding the angle from each camera to its direction. This phase contains 4 main parts :

- 1) Isolation of the object from the background. That is done by comparing the present picture with a static background picture of the room. This enable us to find the objects which are different from the background.
- 2) Separating between different objects and choosing the object we focus on. This is done by scanning the differences picture we got in the previous part, finding groups of pixels classifying them to objects. For this implementation we focus on the biggest object.
- 3) Finding the distortion of the camera's lens and fixing the location of the object accordingly. The distorted picture which was pictured by the camera was compared with the original picture, and the coefficients of the camera's distortion were found. By those coefficients the location of the object can be fixed.
- 4) Calculating the angle towards the object from its location in the picture. A picture in a known length is photographed, thus the number of pixels for each degree in the picture can be found. By this number the object location can be transformed to its angle.

We had to stand in requirements of scaleability and compatibility in order to enable coping with a changes in the environment's conditions and the camera's location. Thus the system modular approach was developed by using a number of modules and Object Oriented Programming (OOP). That's in order to ease changes executions

and supporting compatibility to a future requirements. In order to enable changes in the location of the cameras and the cameras parameters influence on the image processing, a Graphical User Interface (GUI) was built which enable easily changing of the parameters during the program execution. Thus we can investigate and find dynamically the optimal parameters.

By this thesis, an alternative security system model was improved with image processing methods. In this model, movement of object is detected and object position is marked on area's map. In my thesis, image processing methods are used to separate the moving object from the picture. Image enhancement is used to improve the appearance or highlight certain details of an image

. Image segmentation is used for the categorisation or classification of elements or structures within an image. Image

manipulation is used to the geometric alteration of an image, including translation, scaling and sheering.



1. INTRODUCTION

Security's importance is growing up each day. Tracking the movement of people and other objects is important in security system concept. Image processing methods are suitable for these systems. It is useful for security of a covered area to locate the moving object on frames which captured by cameras and mark its route on the area's map.

The thesis's goal is to design and implement a tracking system for moving object in a room. The system recognizes an object and marks its location and route on the room's map. In order to determine the absolute position of moving object, two cameras are used. Each camera "sees" only a two-dimension picture while the map shown by the application represents the third dimension – the depth dimension. Third dimension can be gotten by using second camera. It is obvious that future systems of human motion analysis will greatly benefit from the use of multiple cameras. The availability of information from multiple cameras can be extremely helpful because the use of multiple cameras not only expands surveillance area, but also provides multiple viewpoints to solve occlusions effectively. Tracking with a single camera easily generates ambiguity due to occlusion or depth. However this may be eliminated from another view. For multi-camera tracking systems, one needs to decide which camera or image to use at each time instant. That is, the coordination and information fusion between cameras are a significant problem. Up to now, the emphasis of the development of VisionWorld has been on the reliability of the recognition performance. One problem is the occlusion of some parts of the human body which may negatively affect the correct pose estimation. Other difficulties are changing illumination conditions and arbitrarily composed background situations which may severely disturb the segmentation phase leading to inaccurate or even faulty results.

Coping with these problems is the second purpose of the body model. Occlusions of body parts are reliably anticipated by the three cameras providing different views of the interacting person. Moreover, failures in the recognition of specific body parts in one of the cameras, which may be related to faulty segmentation or pattern recognition results, can be recognized and corrected due to the redundancy provided by the different camera views and the time coherence of observed movements.

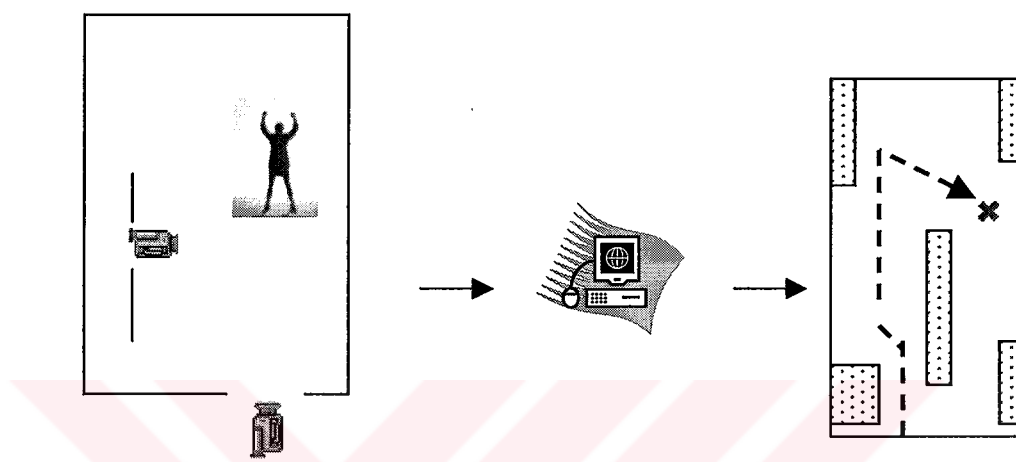


Figure 1.1 System definition

Identifying moving objects from a video sequence is a fundamental and critical task in many computer-vision applications. A common approach is to perform background subtraction, which identifies moving objects from the portion of a video frame that differs significantly from a background model. There are many challenges in developing a good background subtraction algorithm. First, it must be robust against changes in illumination. Second, it should avoid detecting non-stationary background objects such as moving leaves, rain, snow, and shadows cast by moving objects. In order to avoid wasting time on the background, we use image segmentation to separate foreground portions of the scene from the background. Image segmentation is the action of any algorithm that separates regions of an image in a way that resembles how a human would naturally perceive them. Since we are interested in motion, a natural approach is to segment those regions of the image that are moving relative to the background. This process is called background subtraction. Our goal in image segmentation is to separate background areas of the image from foreground regions of motion that are of interest for human tracking. In this project, we make the fundamental assumption that the background will remain stationary. The algorithm of choice is background subtraction, in which we compute a model of

the image background over time. For any given frame of video, we can subtract this background image from it. Those pixels with a result near zero are treated as background and those pixels with a larger result are treated as foreground. Thus, once we have the model of the background image, this algorithm is simple, efficient, and easy to implement.

Nearly every system of vision-based human motion analysis starts with human detection. Human detection aims at segmenting regions corresponding to people from the rest of an image. It is a significant issue in a human motion analysis system since the subsequent processes such as tracking and action recognition are greatly dependent on it. This process usually involves motion segmentation and object classification. Motion segmentation in video sequences is known to be a significant and difficult problem, which aims at detecting regions corresponding to moving objects such as vehicles and people in natural scenes. Detecting moving blobs provides a focus of attention for later processes such as tracking and activity analysis because only those changing pixels need be considered. However, changes from weather, illumination, shadow and repetitive motion from clutter make motion segmentation difficult to process quickly and reliably. At present, most segmentation methods use either temporal or spatial information of the images.

2. COLOR DEFINITION

Almost every human being is very familiar with color, yet it is difficult to define what it really is. Although color is very real and concrete, we are compelled to reach for abstract terms when we want to grasp its nature.

We talk about color when we have a particular sensation in the brain, caused when light radiation of a certain wavelength reaches our eyes. This definition is built up from two parts that are quite different in kind. The first one is of a psychological nature. It deals with the way the sensation of color is processed by the mind. The second one is merely the eyes detection of physical radiant energy. So, color is in fact a psycho-physical phenomenon. It interrelates both a psychological and physical process [1,2].

In 1666 Isaac Newton discovered that white sunlight is composed of all the colors of the rainbow. Physically spoken, this means that white sunlight is a mixture of several electromagnetic rays with different wavelengths. Every color corresponds to another wavelength. The complete visible light spectrum, as in the figure below, ranges from 380 nm (violet) to 780 nm (red).

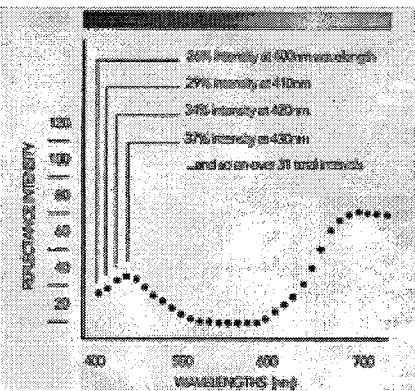


Figure 2.1 Wavelength diagram

We perceive a tomato as red and a lemon as yellow. This perception not only depends on the spectral characteristics of the light source and the human eye, but especially on those of the object itself.

The color of an object is due to the characteristic that it absorbs light from certain wavelengths and reflects light from others. The reflected light from different wavelengths results in the object's color. The tomato absorbs greenish and bluish light while it reflects reddish light.

This characteristic can be described in a spectral reflectance curve, showing the relative intensity of the reflected light at each discrete wavelength.[3]

Just like any physical light source and any physical object, every light detector is more sensitive to certain wavelengths than to others. This can also be expressed in a spectral response curve. The detector we are especially interested in is the human eye.

Experiments have shown that every color sensation can be created by exciting two or three types of cones simultaneously. E.g., we perceive a color as magenta when both the red and blue cones are excited. This is the principle on which colorimetry is based, and is called the additive tri-stimulus theory. It states that a wide range of colors can be matched by adding different amounts of three primary colors. Color displays are also based on this theory. The additive primary colors are red, green and blue.

In 1931 the CIE began to describe color in a quantitative, mathematical way. The basis of this work was formed by two scientists named Wright and Guild who independently performed experiments on human color matching in the 1920s.

2.1. Classification Of Colors

Computer display typically have three electron guns that illuminate three different kinds of phosphor dots on the screen. Those phosphor dots, when illuminated, glow with the red, green, and blue primary colors of light. From these primary colors, all other shades are generated.

A combination of real world physical characteristics determines what the human vision system perceives as color. A color space is a mathematical representation of these characteristics. Color Spaces are always three-dimensional. There are many possible color space definitions:

- Digital imagery often uses the red/green/blue color space, known simply as RGB.
- The cyan/yellow/magenta space, known as CYM, is used in printing.
- Hue, saturation, and intensity, (or HSI) is the color space typically used by artists.
- Intensity-chromaticity color spaces, YUV and YIQ, are used for television broadcast.

Though most work in digital imagery is performed in RGB, which is native to computer displays, many digital image processing applications require transformation to the other color spaces. Usually utility programs are provided to perform these color space transformations [3,4].

Red, Green, Blue (RGB):

All color spaces are three-dimensional orthogonal coordinate systems, meaning that there are three axes (in this case the red, green, and blue color intensities) that are perpendicular to one another. This color space is illustrated below. The red intensity starts at zero at the origin and increases along one of the axes. Similarly, green and blue intensities start at the origin and increase along their axes. Because each color can only have values between zero and some maximum intensity (255 for 8-bit depth), the resulting structure is the cube. We can define any color simply by giving its red, green, and blue values, or coordinates, within the color cube. These coordinates are usually represented as an *ordered triplet* - the red, green, and blue intensity values enclosed within parentheses as shown below.

Several colors are shown mapped into their locations in the RGB cube, or color space. Black has zero intensities in red, green, or blue, so it has the coordinates (0,0,0). At the opposite corner of the color cube, white has maximum intensities of each color, or (255,255,255). Full intensity red, having zero green or blue components, also is positioned at a corner of the cube at location (255,0,0). Yellow, which is combination of red and green, is positioned at (255,255,0). Cyan and magenta, which are combinations of green and blue and red and blue, respectively, are at (0,255,255) and (255,0,255). All other colors can be described by specifying their coordinates within this cube [5].

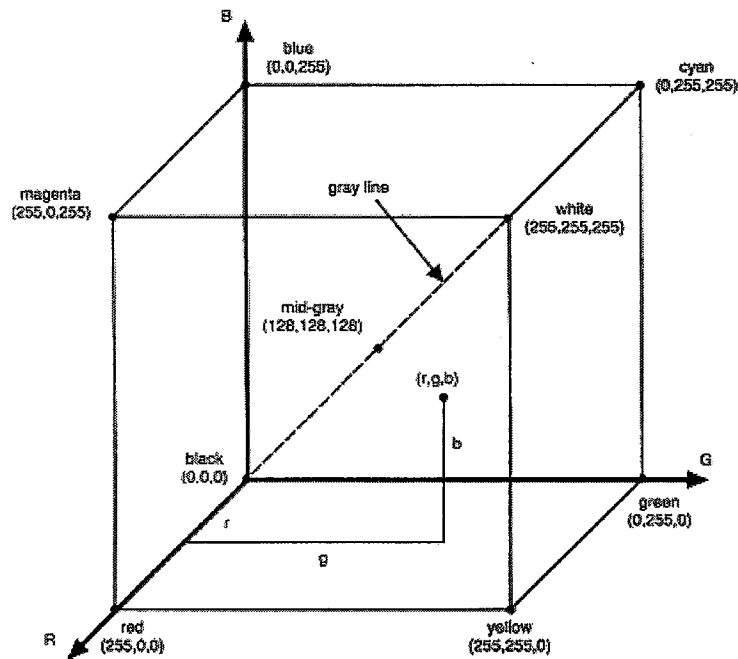


Figure 2.2 RGB diagram

Hue, Saturation, Intensity (HSI)

We see things as colors, or hues, that either have a washed-out look or have deep, rich tones. This means having low or high saturation, respectively. A bright afternoon sun gives everything a high-intensity look, while dusk provides dark images of low intensity.

HSI is very different three-dimensional color space from RGB or CYM. The following figure illustrates a common representation of this space. The cone shape has one central axis representing intensity. Along this axis are all the gray values, with black at the pointed end of the cone and white at its base. The greater the distance along this line from the pointed end, or origin, the brighter or higher the intensity.

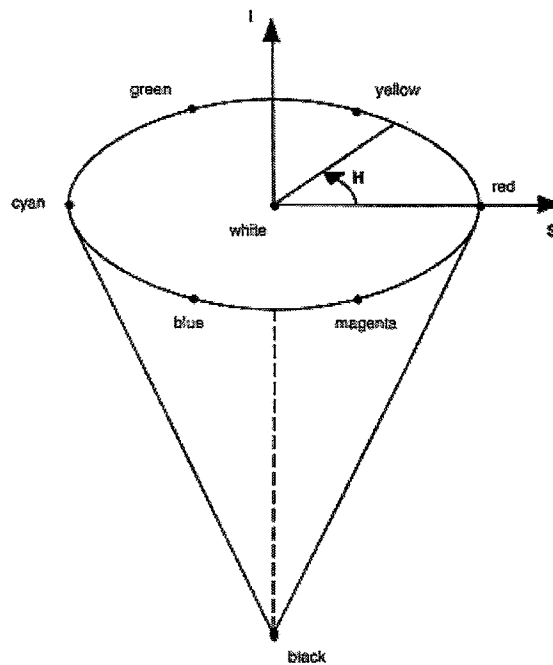


Figure 2.3 HSI diagram

If this cone is viewed from above, it becomes a circle. Different colors, or hues, are arranged around this circle. Hues are determined by their angular location on this wheel. Saturation, or the richness of color, is defined as the distance perpendicular to the intensity axis. Colors near the central axis have low saturation and look pastel. Colors near the surface of the cone have high saturation. [5]

It may be preferable to modify an image in HSI color space rather than RGB. For example, we might want to change the color of a bright yellow car moving down a road to blue, but we want to leave the rest of the scene, including highlights and shadows on the car, unaffected. This would be a difficult task in RGB, but it is relatively simple in HSI. Because the yellow pixels of the car have a specific range of hue, regardless of intensity or saturation, those pixels can be isolated easily and their hue component modified, thus giving a different-colored car.

Since almost all digital image processing systems operate on RGB images, the example described above would be performed in three steps. First, the original RGB image would be converted to HSI. Second, the hue (or saturation or intensity) would be modified. Finally, the image would be converted back to RGB. There are many techniques for transforming an image from RGB to HSI and back again because the transformation is somewhat subjective.

3. DIGITAL IMAGE PROCESSING

Digital Image Processing (DIP) is concerned with the manipulation and analysis of images discretised from continuous signals. Algorithms to manipulate these images mathematically involve integrals, however the necessity of sampling these images for computer use simplifies these formulae in terms of replacing integration operations by summation operations.

The manipulation of these images can be of many forms. The main goals in classical image processing are:

- *image enhancement*, to improve the appearance or highlight certain details of an image,
- *image segmentation*, the categorisation or classification of elements or structures within an image.
- *image manipulation*, a general term which we will restrict the use of to specifically refer to the geometric alteration of an image, including translation, scaling and sheering.

The following sections describes the image processing algorithms and methods that are used on implementation of my thesis. We should concentrate mainly on *point processes* and *area processes* methods [6,7].

Point processes are algorithms to modify images on a pixel by pixel basis. Each pixel is replaced by a new pixel whose value depends only on the current pixel's value, and / or its location. Point processes are the simplest form of image processing algorithm, requiring only a single pass over the image data.

Area processes transform an image, on a pixel by pixel basis, giving each pixel a new value depending on the values, and / or positions, of its neighbours. The neighbourhood of a pixel forms a N by N grid around the pixel, where N is an odd number. Area processes require only a single pass over the image data, but are still

more computationally expensive than point processes, as the number of pixels read to calculate each new pixel value is N^2 times greater than an equivalent point process.

3.1. Moving Object Recognition Techniques

Appearance-based object recognition methods have recently demonstrated good performance on a variety of problems. However, many of these methods either require good whole-object segmentation, which severely limits their performance in the presence of clutter, occlusion, or background changes; or utilize simple conjunctions of low-level features, which causes crosstalk problems as the number of objects is increased. We are investigating an appearance-based object recognition system using a keyed, multi-level context representation, that ameliorates many of these problems, and can be used with complex, curved shapes. Pictures on this page are from a training database we have used in system tests. There are some object recognition techniques.

3.1.1. Background difference technique

This method contain 4 main parts :

I. Isolation of the object from the background. That is done by comparing the present picture with a static background picture of the room. This enable us to find the objects which are different from the background.

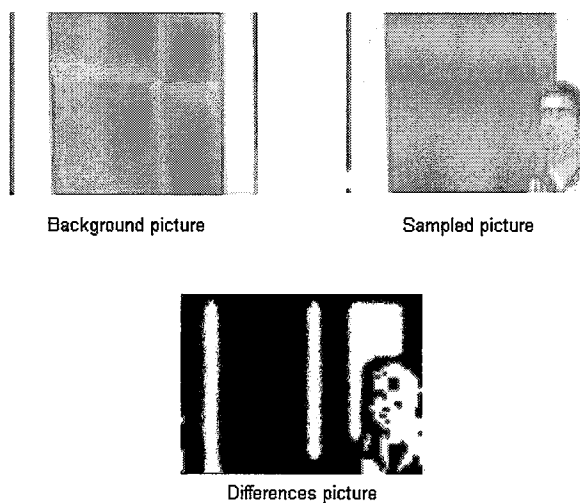


Figure 3.1 Background difference

II. Separating between different objects and choosing the object we focus on. This is done by scanning the differences picture we got in the previous part, finding groups of pixels classifying them to objects. For this implementation we focus on the biggest object.

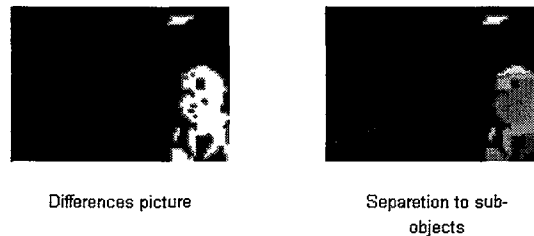


Figure 3.2 Separation the sub objects

III. Finding the distortion of the camera's lens and fixing the location of the object accordingly. By comparing the distorted picture which was pictured by the camera with the original picture, the coefficients of the camera's distortion is found. By those coefficients we can fix the location of the object.

IV. Calculating the angle towards the object from its location in the picture. By photographing a picture in a known length thus the number of pixels for each degree in the picture is found. By this number we can transform the object location to its angle.[7]

3.1.2. Corresponding pixel comparing technique

This method contain 5 main parts :

- I. The original picture is taken from the video film as a single frame from the avi file.
- II. Applied a gaussian filter on the original picture in order to remove noise, added by sampling, quantization, etc.
- III. A sequence of frames are selected and analyzed by the median filter. The median filter was applied in the forth dimension which is the time dimension. Each pixel was compared to the corresponding pixel in the other frames and the median value is selected. The result contains only the background, which is the static part of the picture.
- IV. After the background image is subtracted, we can detect moving objects. By usage of Sobel filter the main edges were detected. The result is a binary image.

V. This stage is constructed from the following steps:

- a. Each 3x3 block that contains more than a certain threshold of ones, was replaced by a single pixel with the value = 1. This is a sort of dilation and sub sampling of the edged picture.
- b. On the picture the actions of dilation and erosion were applied.
- c. In order to receive smooth and complete shapes, we apply one more filtering: using `bwmorph` command with a majority parameter.

After these steps, the larger shapes were detected (above a specified threshold) and their locations were saved.

Due to advantages (more useful, flexible, less process count) of first method, “Background Difference” is selected. In the thesis, “Background Difference” method steps are applied.

3.2. Image Cleaning Techniques

Image histograms form the basis of algorithms that change the dynamic range of pixel values, such as contrast stretching or histogram equalisation. An image histogram is a graph of pixel-intensity distribution, showing the dynamic range of pixel values for that image. Specifically, it plots the number of occurrences of each pixel value in the image, against pixel value. The histogram is usually displayed relative to the largest number of occurrences of any pixel value [8].

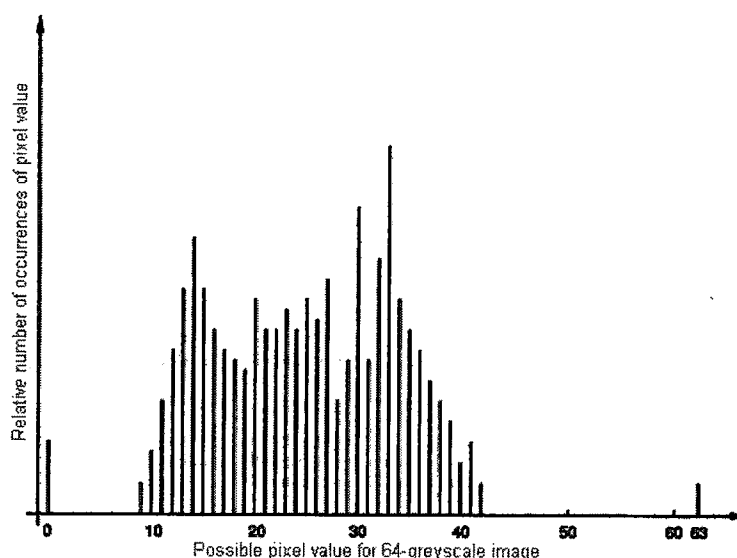


Figure 3.3 64 grayscale histogram

An image with the above pixel-intensity distribution would show a lack of contrast and this may make it difficult to distinguish many details. The point process of *contrast stretching*, described below, would spread the image histogram, widening the pixel-intensity distribution of the image, apparently increasing its level of detail.

Image negation is to make an image resemble a photographic negative. All the light regions of an image should be made dark, and all the dark regions should be made light. In practice this can be achieved by stepping through the image pixel by pixel, subtracting the current pixel value from the maximum possible pixel value. The result of this operation becomes the new pixel value. Alternatively, image negation can be achieved with the use of a lookup table. A lookup table is created, where the value of each element in the lookup table is equal to its index subtracted from the maximum pixel value. For a 256-grayscale image, the first entry in the lookup table would contain the pixel value 255, and the last entry would contain the pixel value 0. The lookup table is then applied to the image, thus negating it.

The object of *brightness adjustment* is to make the image lighter or darker overall by some factor. This can be achieved by stepping through the image pixel by pixel, incrementing the pixel value by a positive constant to make the image lighter, or by a negative constant to make it darker. We must care should be taken to not let any pixel value become greater than the maximum or less than the minimum allowed pixel value. Once again, as an alternative, this algorithm can also be implemented with the use of a lookup table. A lookup table is created, where the value of each element in the lookup table is equal to its index incremented by the (positive or negative) adjustment factor, taking care not to become either greater than the maximum, or less than the minimum allowed pixel value. The lookup table is then applied to the image.

The *contrast stretching* algorithm given in, aims to change the dynamic range of pixel values in an image by brightening pixel values above a certain threshold, using a logarithmic scale. The limiting case is equivalent to *thresholding* an image, i.e. making all pixel values above a given threshold, white - the maximum possible pixel value, and all pixel values below this threshold, black - the minimum possible pixel value.

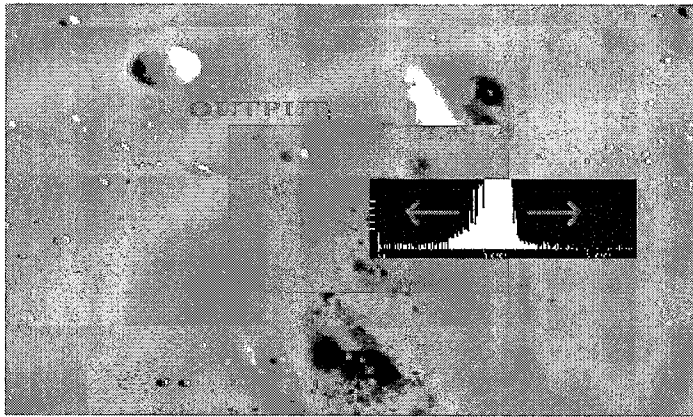


Figure 3.4 Contrast stretching, normalization



Figure 3.5 Adjusting the image histogram to improve image contrast

Histogram equalization is similar to contrast stretching in that it attempts to increase the dynamic range of the pixel values in an image. Unlike contrast stretching, there is no provision for interactivity, and applying a histogram equalization algorithm to an image will always yield the same result.

The image histogram can also be thought of as a chart of the likelihood of a certain pixel value occurring at any point in the image, which we call a probability table. In practice, to convert number of occurrences to probability of occurrence, the image histogram data should be normalised between 0 and 1. This is achieved by dividing the number of occurrences of each pixel value by the total number of pixels in the image (65536 pixels for a 256 by 256 image). The result of this is that the sum of all the histogram data adds up to 1.0.

The next step is to create a lookup table where the n^{th} value in the table is equal to the sum of the first n values in the probability table. This is called the cumulative histogram. Finally each item in the lookup table should be normalised between the minimum and maximum pixel values, and rounded to the nearest integer [8].

3.2.1. Convolution

Convolution is not a complete image processing algorithm in its own right, but rather the core of a very general purpose algorithm, applying a *convolution kernel* to an image. A convolution kernel is an N by M array of weights, where N and M are odd numbers, forming the pixel neighbourhood of each pixel in the image. It is applied by multiplying each value in the pixel neighbourhood by the corresponding weight in the convolution kernel. The sum of these multiplications makes the new value at the centre of the pixel neighbourhood. This value can optionally be divided by a scale factor to make it fit between the minimum and maximum pixel values.

The effect of convolution depends entirely on the weights in the convolution kernel. The individual proofs for many of these kernels require understanding complex mathematics beyond the scope of this tutorial, and are discussed at length in many mathematics textbooks.

3.2.2. Neighbourhood Averaging

Neighbourhood averaging (Mean Filter) is perhaps the simplest convolution kernel process. Neighbourhood averaging is a smoothing process, performed by setting the value of each pixel in an image to the arithmetic mean of all the pixels in its neighbourhood. If the pixel neighbourhood is of size N by M , a convolution kernel of $N \times M$ weights is used, each set to 1, and the sum is divided by a scale factor of $N \times M$. Repeated application of the neighbourhood averaging filter, or use of a larger neighbourhood, causes a greater degree of smoothing. Below are three different sizes of neighbourhood averaging kernel:

- 3 by 3 kernel requiring a scale factor of 9
- 5 by 5 kernel requiring a scale factor of 25
- 7 by 7 kernel requiring a scale factor of 49

1	1	1		1	1	1	1	1		1	1	1	1	1	1	1
1	1	1		1	1	1	1	1		1	1	1	1	1	1	1
1	1	1		1	1	1	1	1		1	1	1	1	1	1	1
				1	1	1	1	1		1	1	1	1	1	1	1
				1	1	1	1	1		1	1	1	1	1	1	1
										1	1	1	1	1	1	1
										1	1	1	1	1	1	1

Figure 3.6 Mean filter kernels

Low pass spatial filters attenuate the high-frequency content of an image, whilst low-frequencies are passed. This can help to remove noise from an image, or allow the low frequency portion of an image to be examined more closely.

Low frequency regions of an image manifest themselves as relatively large areas of uniform colour, or areas in which colour does not change rapidly. Applying kernel (a) below to such a region would cause the pixel value at the centre of the neighbourhood to become the average of all the pixels in the neighbourhood. Since most or all of the neighbouring values would be the same, the centre pixel would be effectively left unchanged.

Regions of high frequency are characterised by rapid changes in colour. Applying the same kernel to such a region would cause fluctuations in pixel intensity to be averaged out, effectively removing, or attenuating, the high frequency content of the image. Below are three low pass filter kernels:

1/9	1/9	1/9		1/10	1/10	1/10		1/16	1/8	1/16
1/9	1/9	1/9		1/10	1/5	1/10		1/8	1/4	1/8
1/9	1/9	1/9		1/10	1/10	1/10		1/16	1/8	1/16

Figure 3.7 Low pass filter kernels

High pass spatial filters attenuate low-frequency image content, whilst passing the high frequencies. It should be noted that in many cases the high pass filters can be thought as behaving more like the electronic equivalent of *active high pass* filters, boosting (or accentuating) high frequency image content, whilst attenuating, or even *passing* low frequencies.

High pass filters, as shown below, are characterised by the large, positive weight at the centre of the pixel neighbourhood, surrounded by smaller, mostly negative weights. The value of the centre pixel dominates the calculation of the new pixel value, whilst the surrounding values help to reduce the effect of the large weight. This gives the effect of accentuating regions of large, rapid change in pixel value, whilst leaving areas of constant pixel value relatively unaffected. Below are three high pass filter kernels:

-1	-1	-1		0	-1	0		1	-2	1
-1	9	-1		-1	5	-1		-2	5	-2
-1	-1	-1		0	-1	0		1	-2	1

Figure 3.8 High pass filter kernels

3.2.3. Median filter

Median filtering serves to remove random noise from an image. It does this by replacing each pixel in an image by the median value from that pixel's neighbourhood. Random noise, which manifests itself as abrupt changes in pixel intensity, would very rarely be the median value in the pixel neighbourhood.

Multiple applications of median filters, or use of large pixel neighbourhood, causes the image to be smoothed.

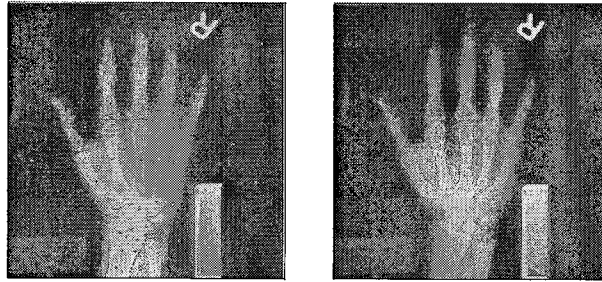


Figure 3.9 Application of the median filter

By calculating the median value of a neighborhood rather than the mean filter, the median filter has two main advantages over the mean filter:

- The median is a more robust average than the mean and so a single very unrepresentative pixel in a neighborhood will not affect the median value significantly.
- Since the median value must actually be the value of one of the pixels in the neighborhood, the median filter does not create new unrealistic pixel values when the filter straddles an edge. For this reason the median filter is much better at preserving sharp edges than the mean filter.

One of the major problems with the median filter is that it is relatively expensive and complex to compute. To find the median it is necessary to sort all the values in the neighborhood into numerical order and this is relatively slow, even with fast sorting algorithms such as *quicksort*. The basic algorithm can, however, be enhanced somewhat for speed. A common technique is to notice that when the neighborhood window is slid across the image, many of the pixels in the window are the same from one step to the next, and the relative ordering of these with each other will obviously not have changed. Clever algorithms make use of this to improve performance.

3.3. Thresholding

In many vision applications, it is useful to be able to separate out the regions of the image corresponding to objects in which we are interested, from the regions of the image that correspond to background. Thresholding often provides an easy and

convenient way to perform this segmentation on the basis of the different intensities or colours in the foreground and background regions of an image.

In addition, it is often useful to be able to see what areas of an image consist of pixels whose values lie within a specified range, or *band* of intensities (or colours). Thresholding can be used for this as well.

The input to a thresholding operation is typically a grayscale or colour image. In the simplest implementation, the output is a binary image representing the segmentation. Black pixels correspond to background and white pixels correspond to foreground (or *vice versa*). In simple implementations, the segmentation is determined by a single parameter known as the *intensity threshold*. In a single pass, each pixel in the image is compared with this threshold. If the pixel's intensity is higher than the threshold, the pixel is set to, say, white, in the output. If it is less than the threshold, it is set to black.

In more sophisticated implementations, multiple thresholds can be specified, so that a *band* of intensity values can be set to white while everything else is set to black. For colour or multi-spectral images, it may be possible to set different thresholds for each colour channel, and so select just those pixels within a specified cuboid in RGB space. Another common variant is to set to black all those pixels corresponding to background, but leave foreground pixels at their original colour/intensity (as opposed to forcing them to white), so that information is not lost.

Not all images can be neatly segmented into foreground and background using simple thresholding. Whether or not an image can be correctly segmented this way can be determined by looking at an intensity histogram of the image. We will consider just a grayscale histogram here, but the extension to colour is trivial.

If it is possible to separate out the foreground of an image on the basis of pixel intensity, then the intensity of pixels within foreground objects must be distinctly different from the intensity of pixels within the background. In this case, we expect to see a distinct peak in the histogram corresponding to foreground objects such that thresholds can be chosen to isolate this peak accordingly. If such a peak does not exist, then it is unlikely that simple thresholding will produce a good segmentation. In this case, *adaptive thresholding* may be a better answer [9].

Figure 3.10 shows some typical histograms along with suitable choices of threshold.

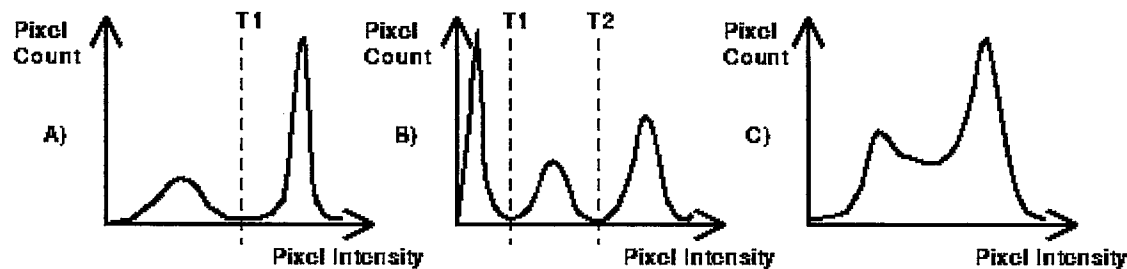


Figure 3.10 A) shows a classic bi-modal intensity distribution. This image can be successfully segmented using a single threshold $T1$. B) is slightly more complicated. Here we suppose the central peak represents the objects we are interested in and so threshold segmentation requires two thresholds: $T1$ and $T2$. In C), the two peaks of a bi-modal distribution have run together and so it is almost certainly not possible to successfully segment this image using a single global threshold.

4. IMPLEMENTATION OF THESIS

The implementation is executed in three basic phases :

- A) Sampling pictures from two video cameras
- B) Image processing
- C) Geometric calculation by the angles and the locations of the cameras

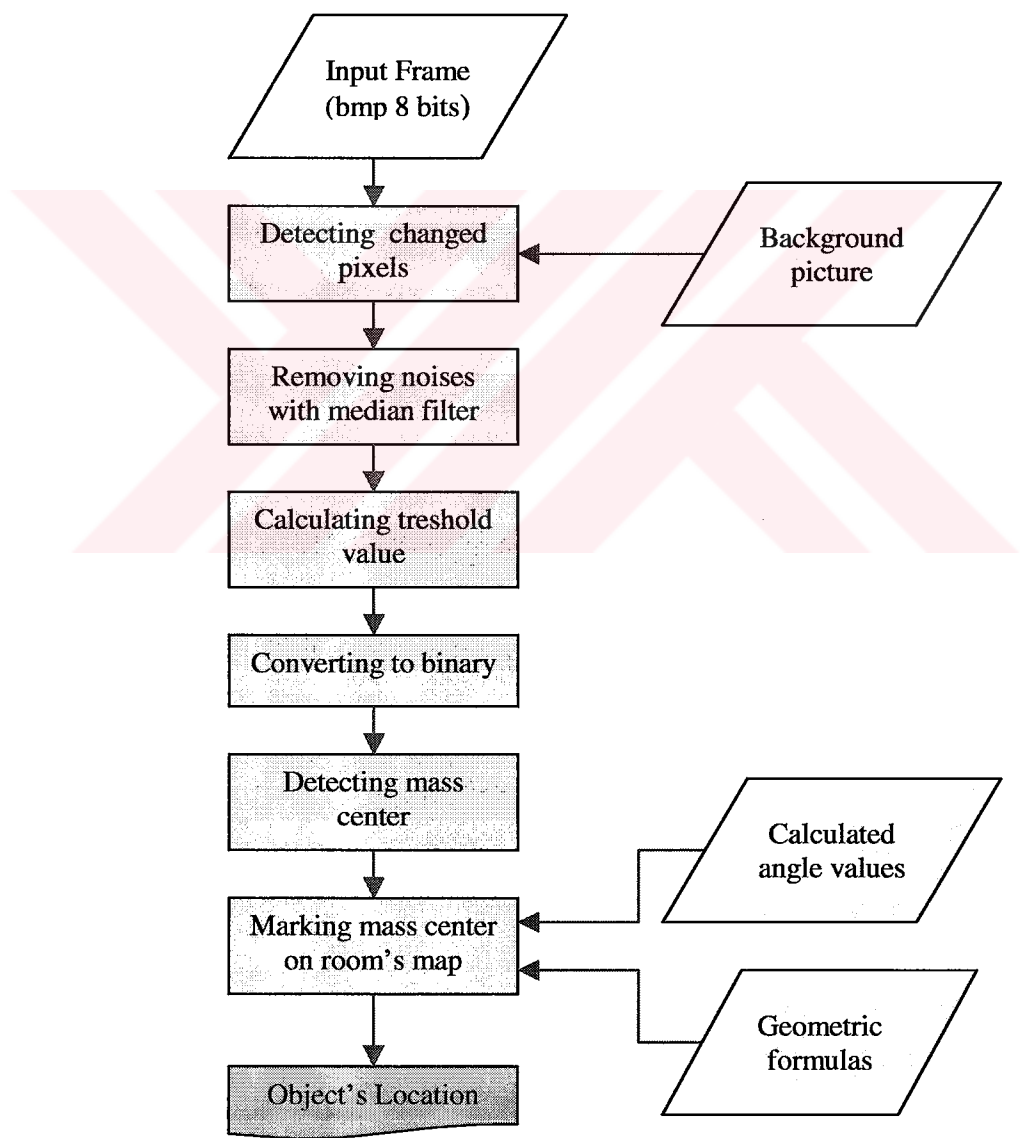


Figure 4.1 Algorithm process

System works on frames which are gotten from videos. A and B camera's each frame is matched with static background image. Thus moving object's position is detected approximately. Absolute position is gotten by applying cleaning filters. Frames are be converted to binary image using the computed treshold value. Geometric transformations and calculations help us to find object's body center and mark this point on room's map. In the next step the interface that supplies the communication between system and user is implemented. All of these steps are explained below in detail.

4.1. Getting Frames

Speed is important in security systems. Because of that it is not necessary to use high quality pictures and fluently video records. Most of these systems use 256 gray-scale color frames and 1-10 frame per second.

There are three different categories of file formats: bitmap, vector, and metafiles. When an image is stored as a bitmap file, its information is stored as a pattern of pixels, or tiny, colored or black and white dots. When an image is stored as a vector file, its information is stored as mathematical data. The metafile format can store an image's information as pixels (i.e., bitmap), mathematical data (i.e., vector), or both.

The Bitmap file format is used for bitmap graphics on the Windows platform only. Unlike other file formats, which store image data from top to bottom and pixels in red/green/blue order, the BMP format stores image data from bottom to top and pixels in blue/green/red order. This means that if memory is tight, BMP graphics will sometimes appear drawn from bottom to top. Compression of BMP files is not supported, so they are usually very large. When saving a file to the BMP format, add the ".bmp " file extension to the end of its file name.

For example; suppose we have 24-bit image 1024x768 (this is a common resolution for satellite images, electronic astral photograpgs and other high resolution graphics.) This may produce a file over 2 megabytes in size. ($1024 \times 768 \times 24 / 8 = 2.359.296$ bytes). All color variations are derivate from three primary colors, Red, Green and Blue. Each primary color is represented by 1 byte (8 bits). 24 bit images use 3 bytes per pixel. If information is stored in LSB (Least Significiant Bit) of each byte, 3 bits can be stored in each pixel. Unfortunately, 24 bit images are uncommon quitey

large. They would draw attention to themselves when being transmitted accross a network. Compression would be beneficial if not necessary to transmit such a file. But file compression may interfere with the storage of information. There are two kinds of compression, Lossless and Lossy. Both methods save storage spaces but may present different results when the information is uncompressed.

Another alternative for 24 bit images, is 256 color BMP images. These are the most common images for image processing methods. Each pixel is represented as a byte (8bit). Gray-scale images are very good, because the shades gradually change from byte to byte. The importance is not whether the image is high quality or not, the importance is recognising the changed pixels.

We assume that, the videos that captured by A and B cameras are saved as 8 bits BMP frames. These frames can be stored in any folder on any path. However, A camera's frames must be saved as PICA0.BMP, PICA1.BMP, PICA2.BMP, PICA3.BMP,... ; B cameras frames must be saved as PICB0.BMP, PICB1.BMP, PICB2.BMP, PICB3.BMP,... in same folder. Background images are gotten by using these frames.

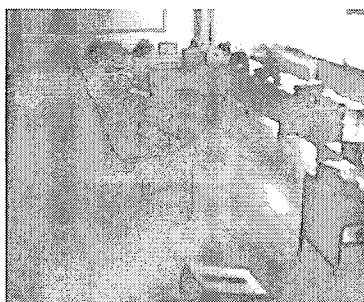
4.2. Comparing Frames With Background

First of all, user must define frames' path. First frames (PICA0.BMP, PICB0.BMP) represent background images. These two picture are loaded to two static arrays. Each 8 bits frame also is loaded to temporary array pixel by pixel when it is shown on interface. Static and temporary arrays include integer numbers 0 up to 255.

Every pixel of frame that pocesing at that time, is subtracted from match pixel on background image. abs function is used for negative results :

$$\text{FrameN}[i][j] = \text{abs}(\text{FrameN}[i][j] - \text{Background}[i][j])$$

Figure 4.2 and Figure 4.3 shows subtraction results.



(a)

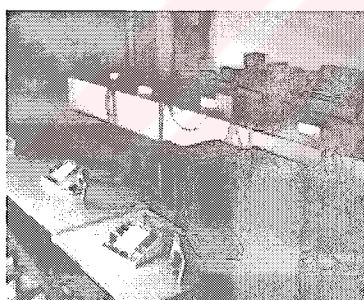


(b)



(c)

Figure 4.2 a-Background image of B camera b-Any frame of B camera
c-After subtraction



(a)



(b)



(c)

Figure 4.3 a-Background image of A camera b-Any frame of A camera
c-After subtraction

4.3. Cleaning Noises On Frames

In some environment, illumination is at high degree. The subtraction images that captured in a high level illuminated environment, includes so much noise. At these conditions, it is too hard to detect the moving object's absolute position. For solving this problem, different cleaning filters can be applied to noisy picture.

Most of filter techniques include complex calculations. Mean and Median filters are preferred after comparison between some cleaning filters according to less complication and process count.

3x3, 5x5 and 7x7 matrix used for main filter application. Figure 4.4 shows results of this filter.

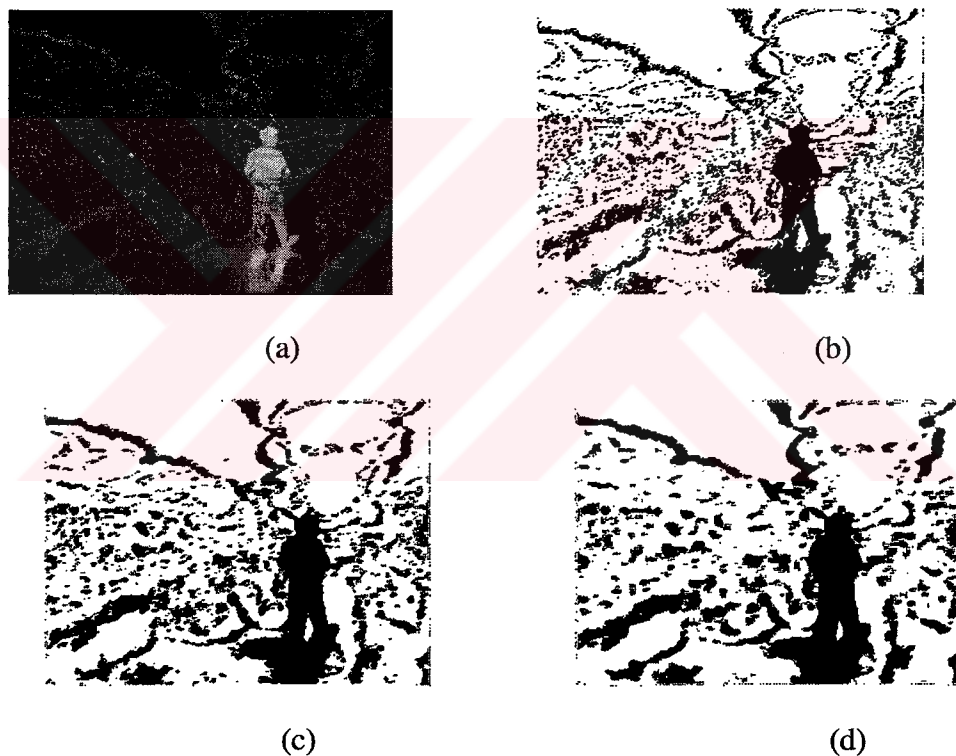


Figure 4.4 **a**-Subtracted image **b**-After mean filter with 3x3 matrix
c-After mean filter with 5x5 matrix **d**-After mean filter with 7x7 matrix

Mean filter used to remove the noises that created by illumination differences. But it didn't gave us perfect results. Even it made more noise on images. For this reason mean filter was eliminated from algorithm.

Median filter applied with 3x3 and 5x5 matrix on frames. Median filter includes sort algorithms. Thus 7x7 or bigger matrix wastes so long time on image applications.

Figure 4.5 shows results of median filter application.

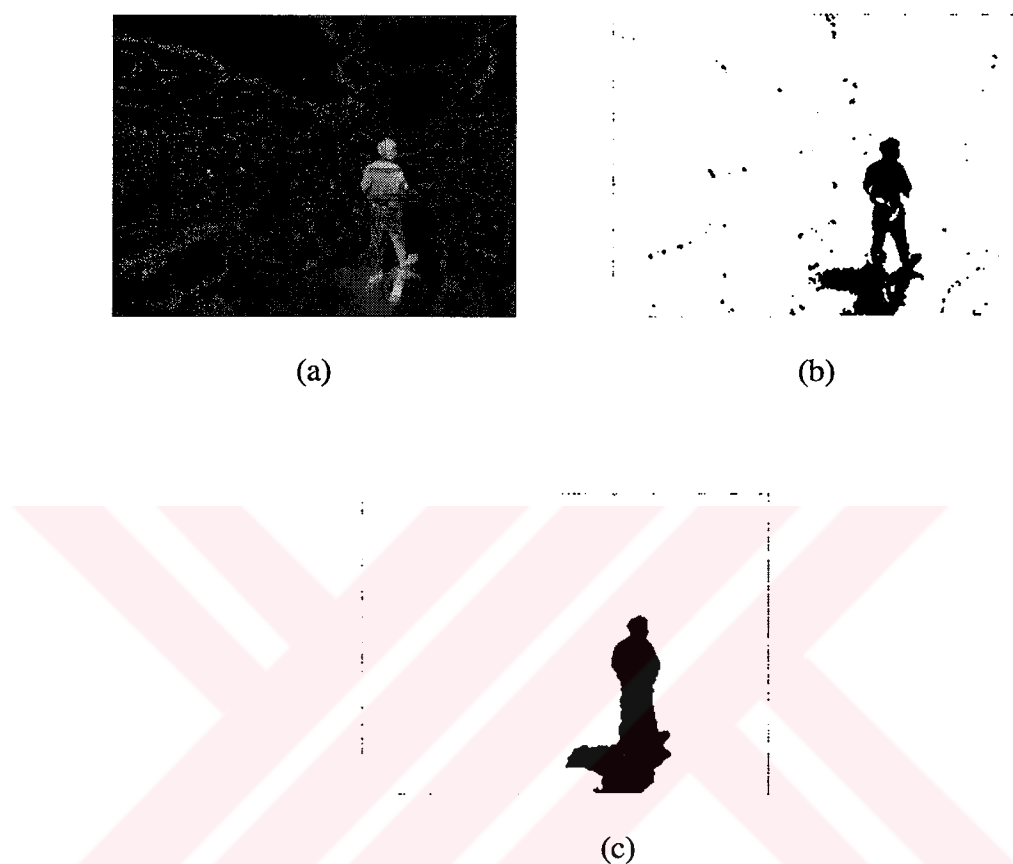


Figure 4.5 a-Subtracted image b-After median filter with 3x3 matrix
c-After median filter with 5x5 matrix

3x3 matrix didn't gave us better results from main filter. But 5x5 matrix gave almost perfect results. Filter was not applied to edge pixels on frame. The noises at edge are be removed in a loop. Moving object's shade which created by illumination and reflection, couldn't removed by this filter.

4.4. Thresholding Frames

The image which filter output must be converted to binary image in order to process on it. The conversion is made by using tresholding techniques. Also a static treshold value can be used but this method decreases flexibility and removes independence of environment (*Static Treshold Calculation*). Treshold value can be changed for every

frame because of color and illumination level difference. So a new threshold value is calculated for every frame (*Dynamic Threshold Calculation*).

In dynamic threshold calculation, content of each pixel will be sum, then sum will be divided by pixel count. In conclusion, absolute threshold value is calculated. Image is converted to binary image by using this threshold value. Results of these process are seems below :

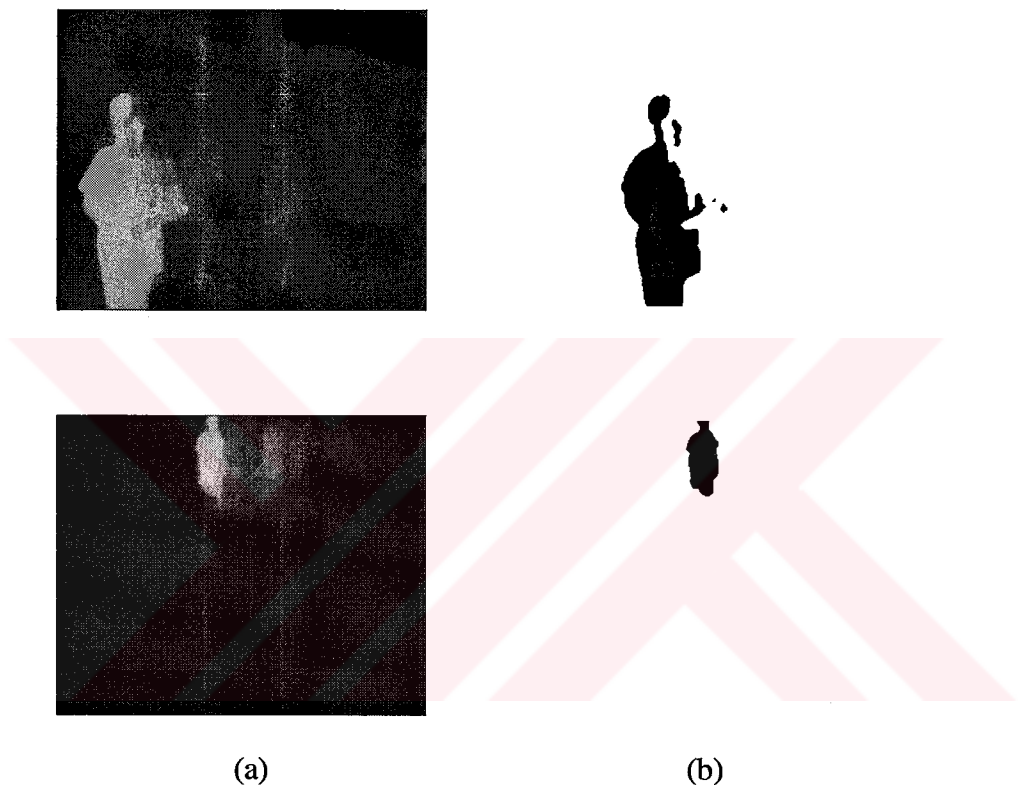


Figure 4.6 **a**-Subtracted image **b**-After threshold

4.5. Calculating Mass Center

Moving object is determined on image after thresholding. In this step, object's mass center is calculated. In the next step, center point will be used for marking object's position on room's map.

There are some kind of mass center calculating methods. A simple method uses sum of lines' and columns' contents. The center is intersection point of the line and column which have biggest sum value. This method finds a mass center point approximately. But it gives wrong value for complex shaped objects. Figure 4.7 shows a wrong calculation.

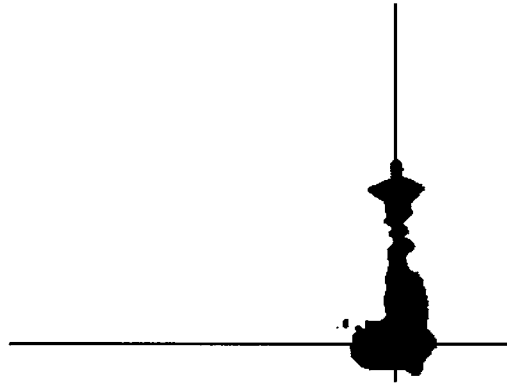


Figure 4.7 A wrong mass center calculation

Another mass center calculation method uses Connected Component technique. It determines moving items on image and finds their center. Then calculates the average center point. But this method works so hard on a fragment flowing system. Also it wastes so much time [10].

My system uses more useful and flexible technique. In this method, each pixel's line and column number will be sum if pixel's value is equal to '1'. Then line and column sum will be divided by sum of '1's. In conclusion we can get absolute mass center. The following pictures shows calculated mass center samples :

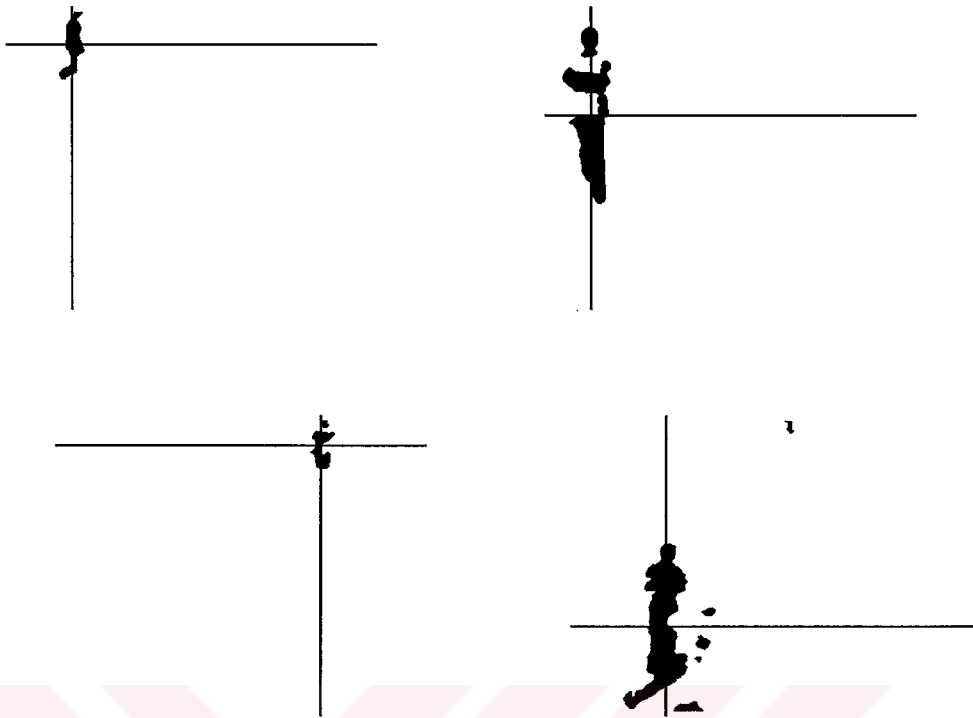


Figure 4.8 Calculated mass centers.

Some frames may not include moving object. So system can't recognize any object and mass center. When working on these frames, mass center will save its old value. Figure 4.9 shows two consecutive frames. First one includes a object but second one doesn't. In the second frame center point and center lines stay at old position.

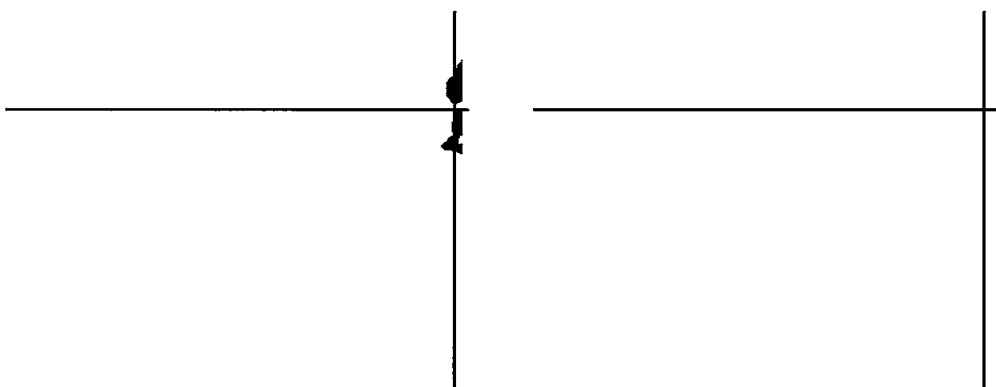


Figure 4.9 Mass center on frame without a moving object

4.6. Geometric Calculation

After calculating mass center, we can measure object's pixel distance from cameras' symmetry line. To mark object's position on map, this pixel value must be converted to centimetre. In order to get this value, we must know number of pixels for each degree in the picture. For calculating this ratio, a picture in a known length is photographed. Figure 4.10 shows measured object.



Figure 4.10 Measured object

In Figure 4.10, length of table and length of picture was measured in centimetre. Pictures width and height are also known in pixel value. Table's distance from camera also measured. This operation is shown in Figure 4.11.

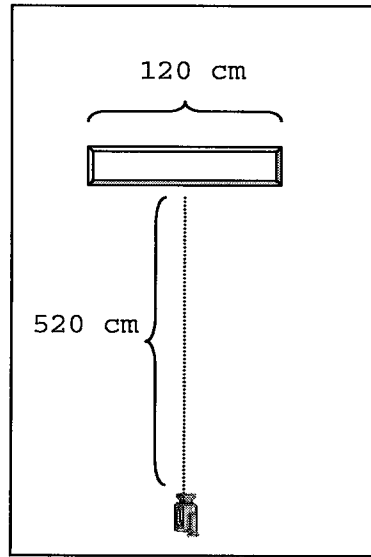


Figure 4.11 Measuring operation.

Number of pixels for each degree in the picture ($\tan(\alpha)$) can be calculated with these values. $\tan(\alpha)$ value isn't depend on environment. This value only depends on cameras' distortion. Most of security cameras use similar distortion value. Because of that, only one measuring operation is enough.

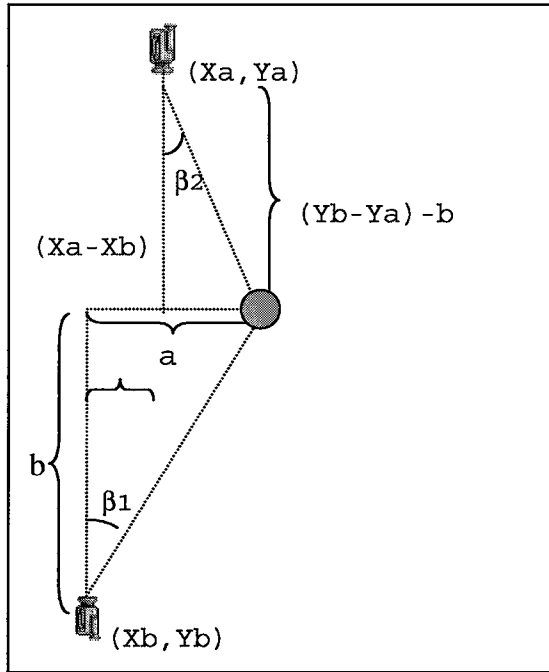


Figure 4.12 Geometric projection

$$a = b \cdot \tan(\beta_1)$$

$$a - (X_a - X_b) = [(Y_b - Y_a) - b] \cdot \tan(\beta_2)$$

$$b = (Y_b - Y_a) \cdot \tan(\beta_2) + (X_a - X_b) / \tan(\beta_1) + \tan(\beta_2)$$

After applying the method above, we can measure $\tan(\alpha)$ and $\tan(\beta)$ using the object's pixel distance from cameras' symmetry line. In order to calculate the object's centimetre distance from cameras, cameras positions have to be used as following formulas :

$\tan(\alpha)$: Measured by using image processing method

$\tan(\beta)$: Measured by using image processing method

X_a : First camera's horizontal position

Y_a : First camera's vertical position

X_b : Second camera's horizontal position

Y_b : Second camera's vertical position

a : Object's horizontal distance from left camera

b : Object's vertical distance from left camera

4.7. Marking The Route

Image processing and geometric calculation methods are applied for every sequential frame. After frame processing, object's absolute position is calculated simultaneously. Object's position on room's map is calculated by using this value. Then this position is marked on map. These processes are applied once for each sequential frame. Then object's route is marked on map.

4.8. Interface Design

A Graphical User Interface (GUI) is designed for communication between users and system. Room and camera parameters are entered to system via this interface. The frames that captured by right and left cameras are shown on this interface consecutively. Frame flowing can be paused and continued by user. The room's map which on the interface are used for mapping the route of the moving object.

The software was developed in C++Builder environment. The thesis was written under the WinXP operating system. The following hardware was used in this project:

- Intel Pentium III CPU
- 128 MB RAM
- 13,2 GB HDD
- 16 MB Graphic Card
- CreativeGoMini WebCam

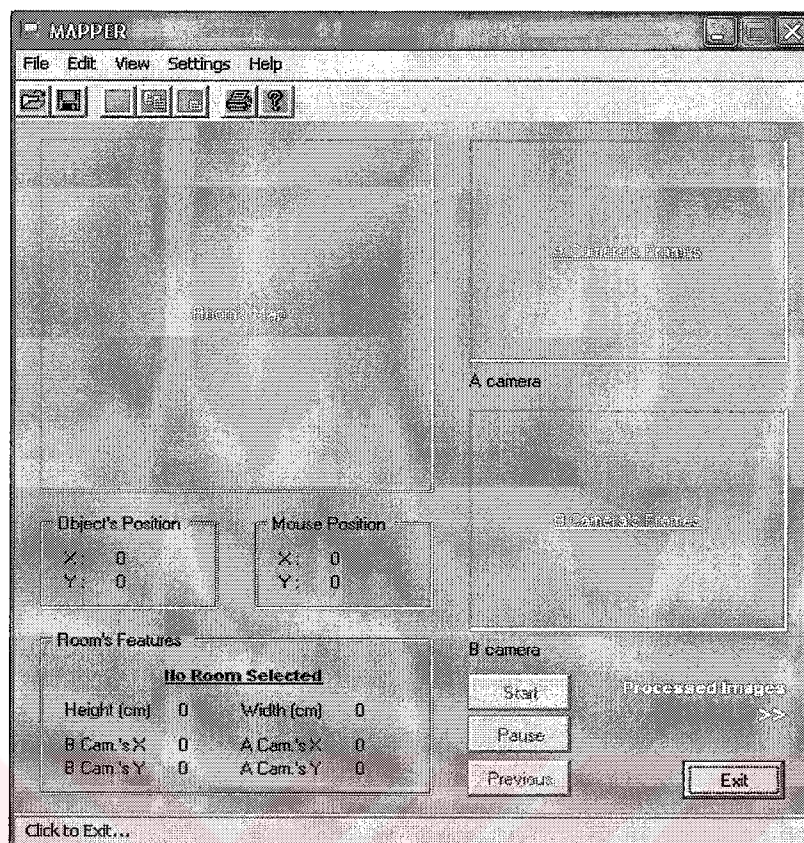


Figure 4.13 Interface window

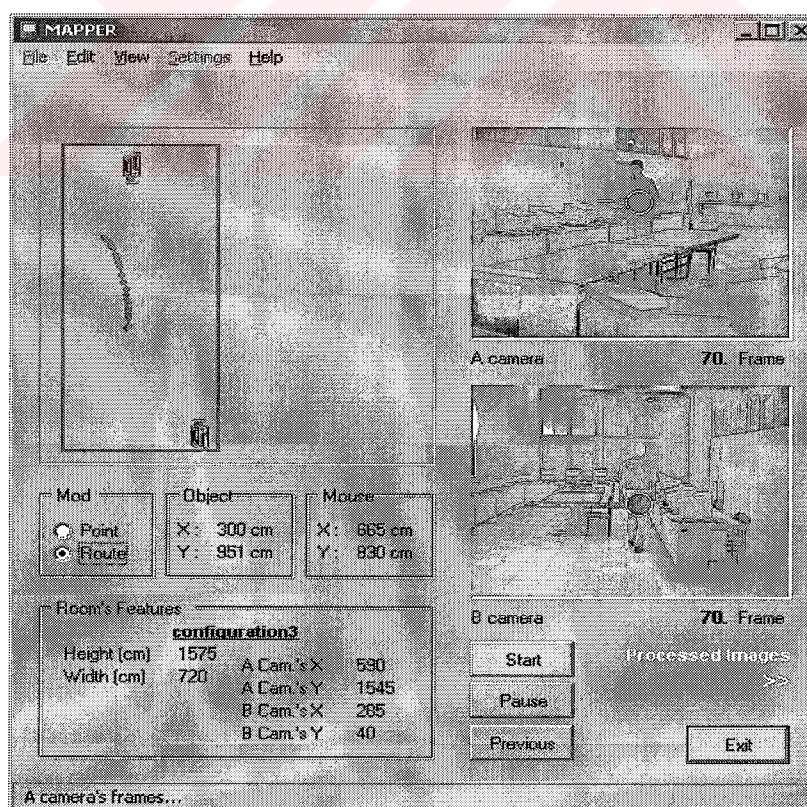


Figure 4.14 Interface window while process is running

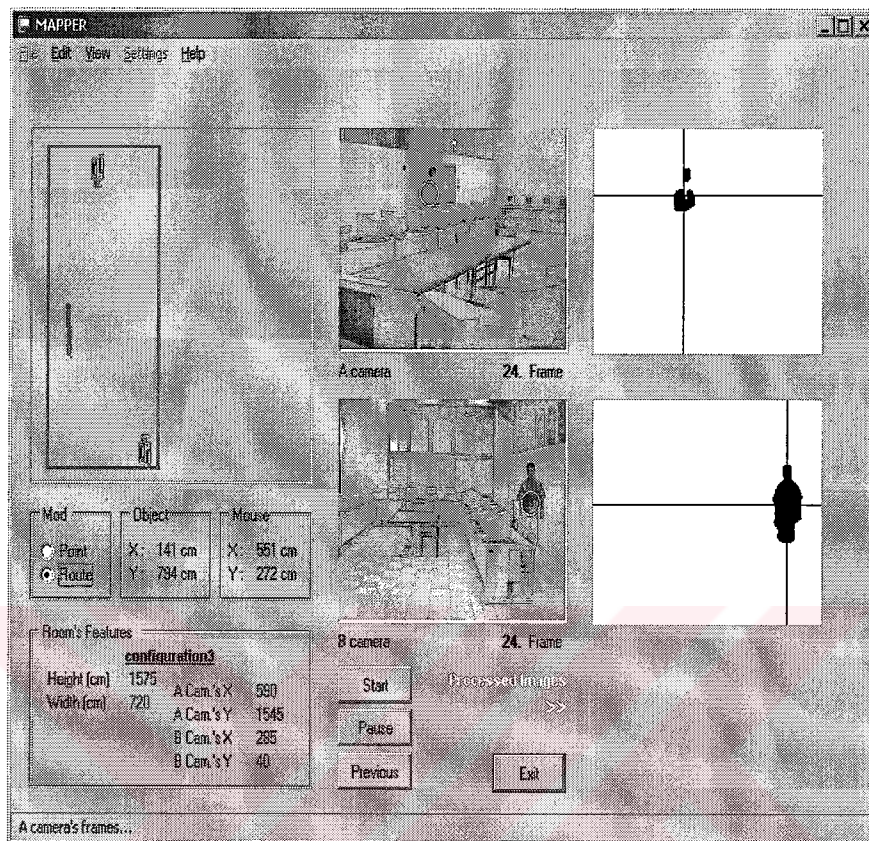


Figure 4.15 Interface window with processed images

Object's location is shown on the map which illustrated in Figure 4.15. Absolute position is determined as centimetres on this figure. In addition, cameras positions are shown on the room's map. Frame flowing and locating can be started by "Start" button and paused by "Pause" button.

In order to enter parameters to system effectively, a menu located at the top of the interface window. This menu includes following items :

- **File**
 - o Open Frame : It selects starting frame.
 - o Save Frame : It saves processed images in 'Saved' folder.
 - o Start : It starts frame flowing and determining of moving object's location.
 - o Pause : It pauses frame flowing and determining of moving object's location.
 - o Print : It prints any pocessed image.

- o Print Setup : It changes printer settings.
- o Exit : It closes program.

This menu illustrated below :

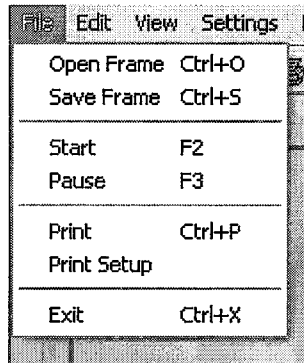


Figure 4.16 'File' menu

- **Edit**

- o Copy : It copies any selected item.
- o Paste : It pastes any item copied from clipboard to anywhere.

This menu illustrated below :

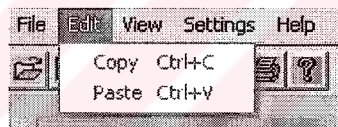


Figure 4.17 'Edit' menu

- **View**

- o Tool Bar : It shows or hides the Tool Bar which includes shortcuts.
- o Status Bar : It shows or hides Status Bar that informes user about running processes.

This menu illustrated below :

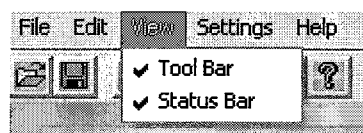


Figure 4.18 'View' menu

- **Settings**

- o Room - Cameras Settings : System stores some room records. In these records, rooms' sizes and cameras' positions are kept. Room - Cameras Settings menu let the user to define new rooms and to change rooms' parameters.
- o Picture Settings: It is used to see frames' width and height and sensibility value.
- o Analysis Settings : It is used to determine 'first' and 'last' frames. System works between defined frames.
- o Flow Control : It is used to control the flow speed and used to determine frame percentage of video.

This menu illustrated below :

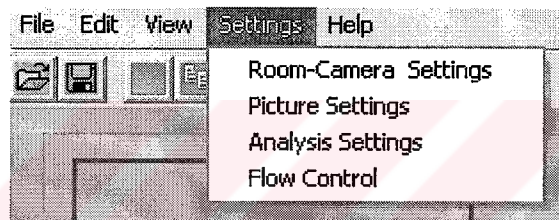


Figure 4.19 'Settings' menu

- **Help**

- o Help : It helps to user.
- o About Mapper : It informs the user about MAPPER.

This menu illustrated below :

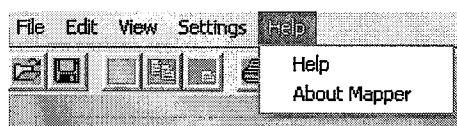


Figure 4.20 'Help' menu

System stores room records. Program can be applied for every record. For every record, room's width, height and cameras' position parametres must be entered. This window can be accesed by using Settings/Room-Cameras Settings path.

User can see frames' size. In the next development steps, frames are converted to Binary (0-1) form. So, we have to use a treshold value for this process. User can define a sensitivity value for tresholding operation. If user don't define, system

calculates best threshold value. This value is obtained by using background frame's histogram. User can see picture and sensibility parameters via this window :

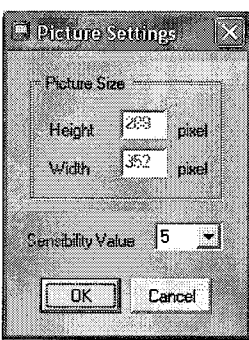


Figure 4.21 'Picture Settings' window

This window can be accessed by using Settings/Picture Settings path. System processes each frame consecutively in the defined path. User can determine first and last frames by using the form below :

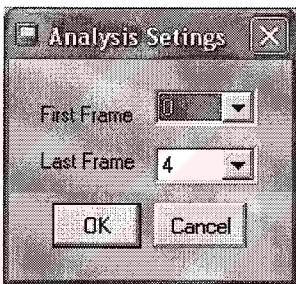


Figure 4.22 'Analysis Settings' window

In Figure 4.23, system processes between Frame0 and Frame4. This window can be accessed by using Settings/Analysis Settings path. User can define the flow speed of frames. He can select 100 up to 4000 (msc). Frames are shown on the form with defined speed value. Default speed value of this properties is 600msc.

Another flow control option is "frame percentage". User can define frequency of video frames. For example, if user selects "5" from list; system shows PICA0, PICA5, PICA10..... and PICB0, PICB5, PICB10..... with this property, user can change flow control. User can see flow control options via this window.

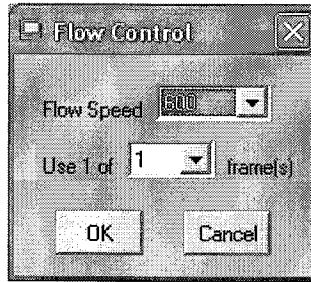


Figure 4.23 'Flow Control' window

Help menu guides the use how to use system interface effectively. User can see help menu via this window :

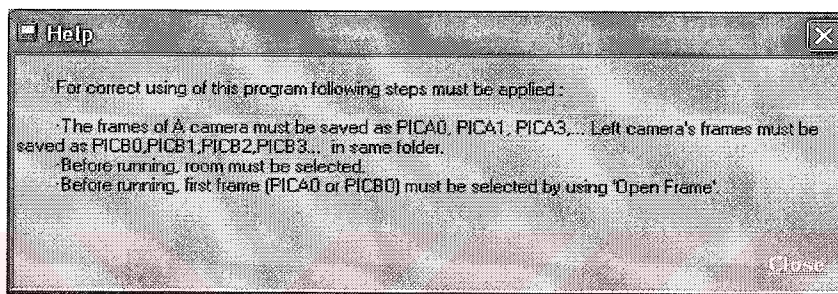


Figure 4.24 'Help Menu' window

The Status Bar which located at the bottom of the main window is used to inform user about running processes.



Figure 4.25 Status Bar

Visibility of Status Bar is depend on user's choice .

Tool Bar which located under the main menu, is used to show shortcuts to program modules . Figure 4.28 shows Tool Bar.

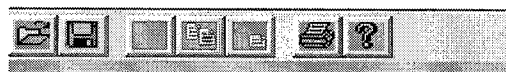


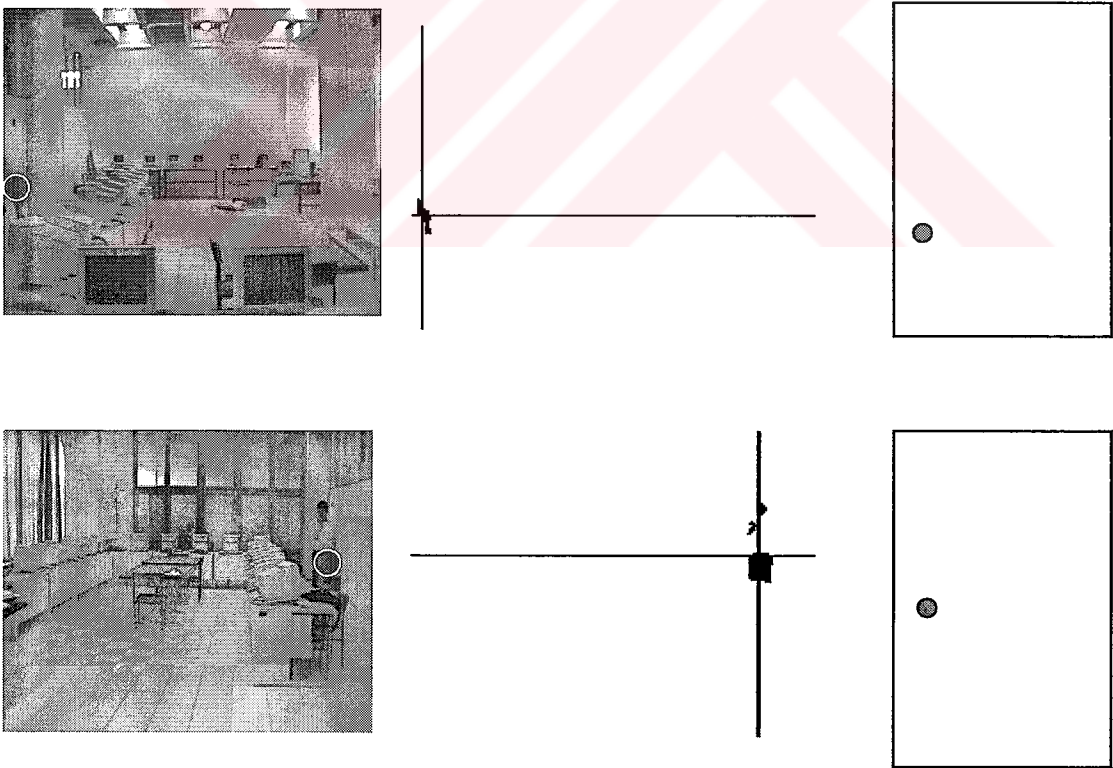
Figure 4.26 Tool Bar

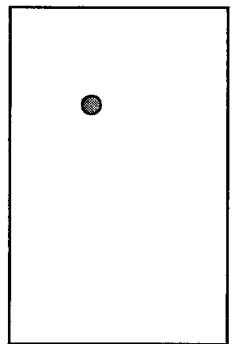
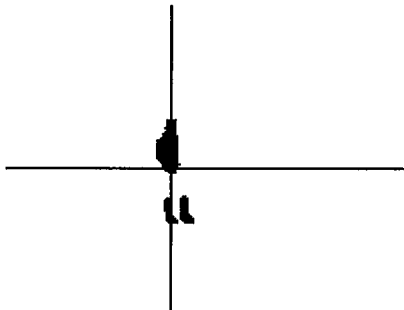
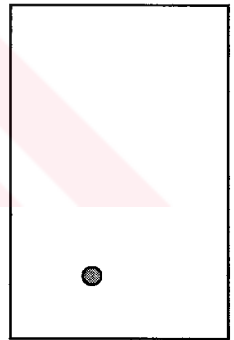
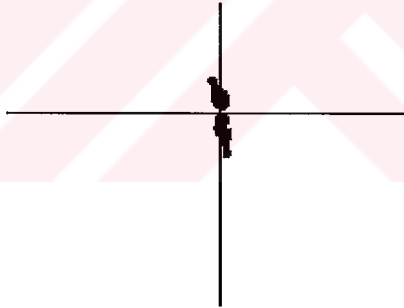
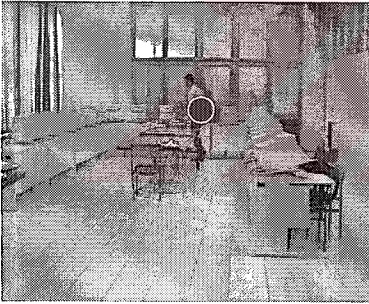
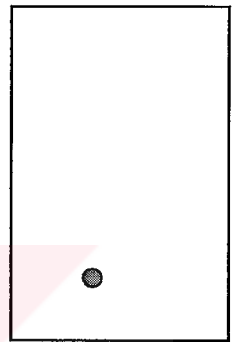
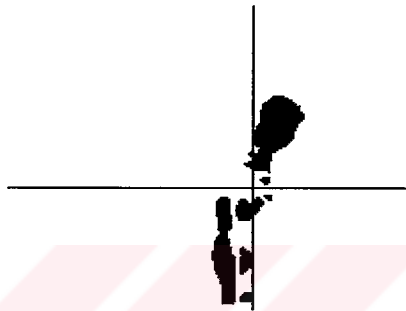
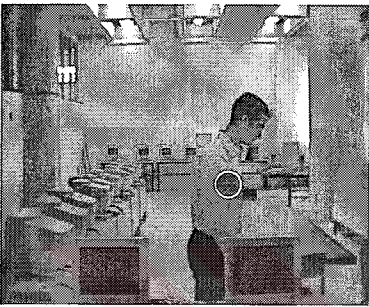
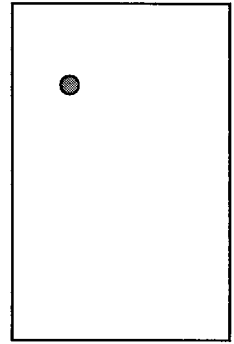
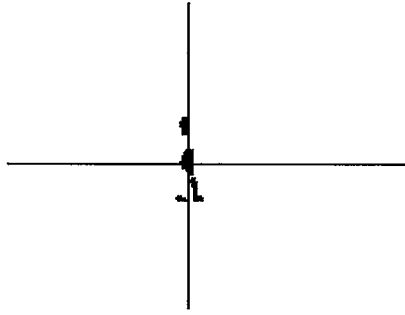
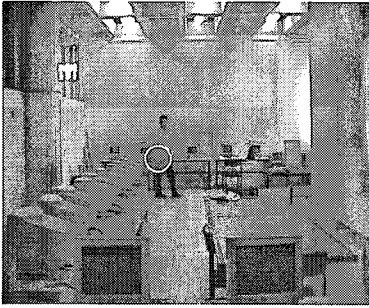
5. EXPERIMENT STATISTICS

As the frames pass, we save the statistics of each frame in order to do a final analysis of the algorithm used. we compare the algorithm success to the actual location of the moving object.

Some kind of environment conditions effect system algorithm. Illumination and reflection may cause wrong results. But most of these problems discarded by flexible designed system. Most of frames gave right result.

The algorithm was tried over 4 videos. The videos differ in the illumination and reflection of the video and in the percentage of frames in which the moving object is hidden or blend in any other static object. Some frame results are shown below.





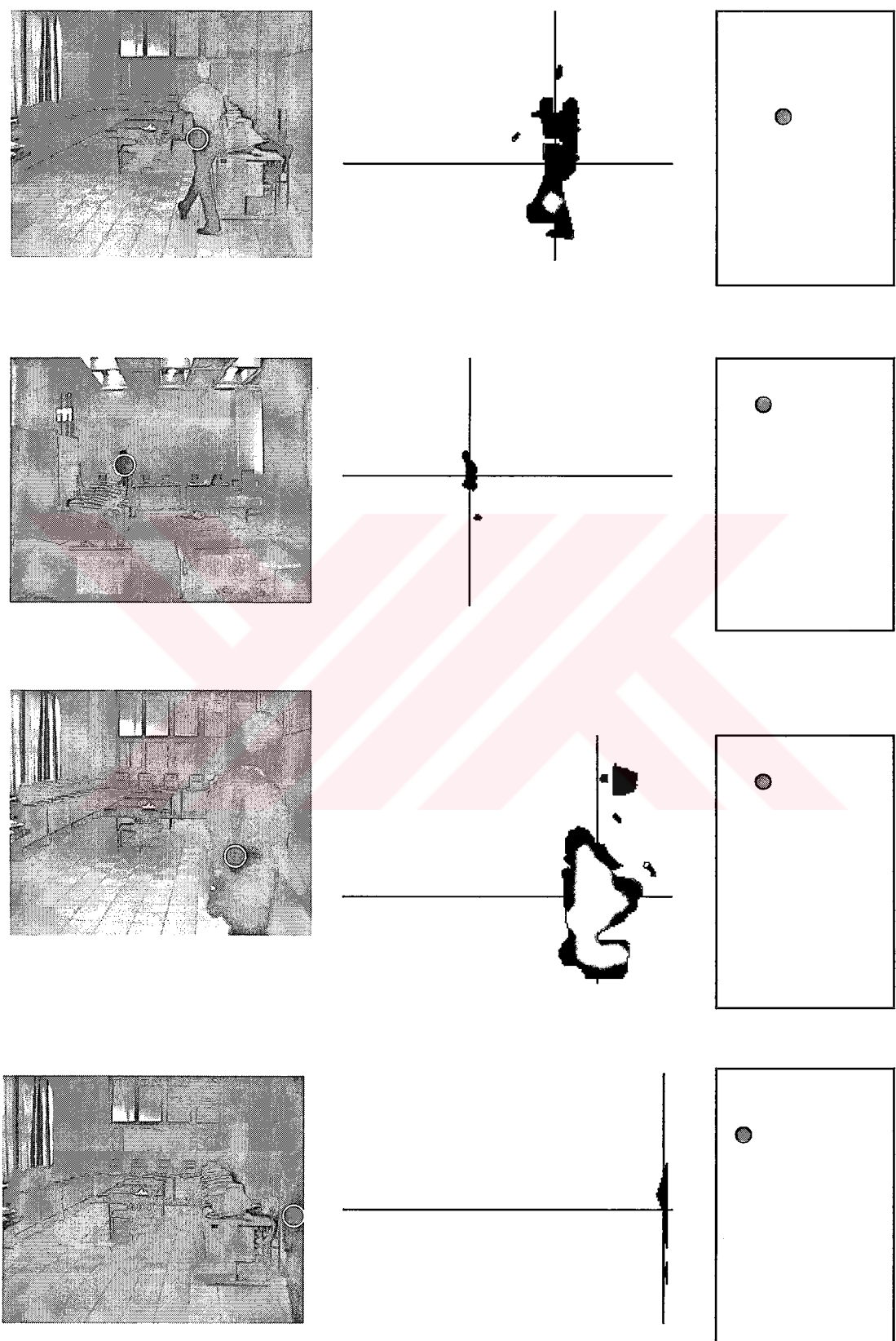


Figure 5.1 Test results

The algorithm was tried over 4 videos. Video 1 is the ‘ideal’ video. The object is never hidden, and its direction has no drastic changes. The second video includes movement of the object between the static objects that causes scenarios of hidden object and changes in the direction. The third video is taken from a larger distance and consequently the object is smaller. The fourth video is longer then the others, and the ball is constantly moves between objects, which causes it to blend in objects many times. The results and statistics are described in the following graph:

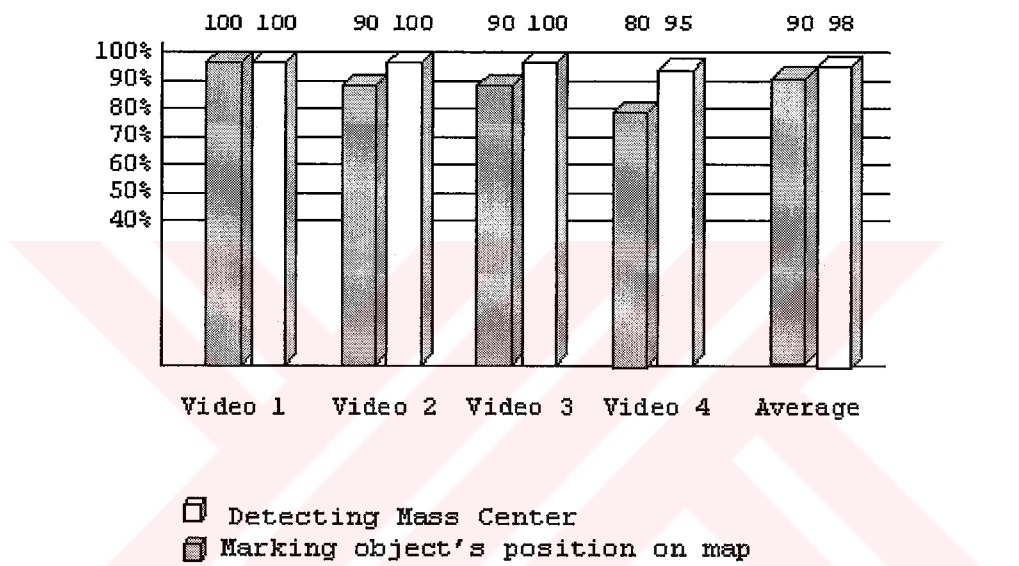


Figure 5.2 Percentage of success

6. SEGMENTATION USING BACKGROUND REGISTRATION

Video motion segmentation algorithm has been proposed based on change-detection.

The algorithm is divided into the following five major steps:

1. Frame difference mask calculation: For each pixel a Frame Difference Mask(FD) is calculated by considering the change in the pixel value for two consecutive frames. If the change in the pixel value is greater than a threshold then that pixel is masked as changing, and stationary otherwise.
2. Background registration: In this step FD calculated in the above step is used for detecting background region. If FD is stationary for several consecutive frames then that pixel is marked as a part of the background. This is done by maintaining a stationary map M of pixels. Every time a pixel is found to be stationary, the count corresponding to that pixel in M is incremented. Thus the value of a pixel in M denotes the extent to which that pixel should belong to background.
3. Background difference: This step generates a Background Difference Mask(BD). This is calculated for each pixel by thresholding the difference between current frame value and the value stored in M . This value will be later used in object detection.
4. Object detection: Using the values calculated in above steps, for each pixel an Object Mask(OM) is calculated under various region descriptions (e.g. Moving, Stationary, Uncovered Background etc.). So, OM assigns one of the regions to each pixel.
5. Post processing: After step 4, an initial OM is generated. But it contains some noise in background and object both. In post processing step, these noise are removed by smoothing the boundary and applying other filters.

Figure 6.1 shows a result of running this background registration technique. Moving object and its shadow have been well segmented from the background.

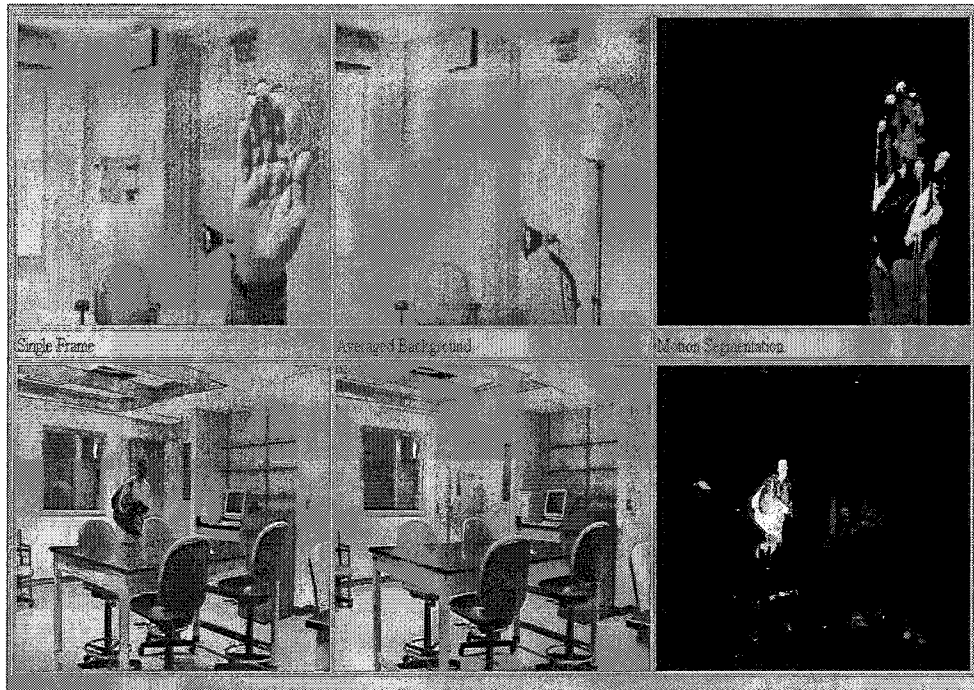


Figure 6.1 Segmentation using background registration technique

6.1. The Algorithm

The segmentation algorithm used by my system is a modified version of the algorithm proposed in "Shyh-Yih et.al. Efficient moving object segmentation algorithm using background registration technique. The algorithm can be described briefly as follows [11].

Data structures used by the algorithm:

2D buffers (arrays) used, each of size of an video frame

- Curr_frame[][] - used to store current frame
- Prev_frame[][] - used to store previous frame
- Bg[][] - used to store the background of the frame
- Bg_count[][] - used to store the count of number of frames for which a pixel has been having the same value.
- Fg_mask[][] - boolean matrix used to mark the pixels identified as foreground.

Thresholds used

- BgThreshold - Threshold for background difference
- FdThreshold - Threshold for frame difference
- CountThreshold - Threshold for background registration

Algorithm Phase I – Initialization:

Initialize all the arrays to 0, except Foreground mask which is initialized to 1 (i.e. initial frame is considered as foreground)

Algorithm Phase II - Obtaining initial foreground mask:

For each pixel in each of the frame do

1. Get the color of the pixel in RGB format
2. Separate the RGB components o find the absolute difference between the color components of the current frame and the previous frame
3. Add all these differences to obtain a frame difference metric
4. If the frame difference is greater than the FD threshold, it means that the pixel is moving. Reset the bgcount for that pixel to 0.
5. If the frame difference is less than the FD threshold, it means that the pixel is stationary for the moment. Increment the bgcount for that pixel by one.
6. If the bgcount increases above count threshold, it means that the pixel has been stationary for a long time and now can be considered as a part of background. Put the pixel value in the background array.
7. Now, obtain the color components of the background for this pixel and find the absolute difference between them and the color components of the current frame.
8. Add all these differences to obtain a background difference metric.
9. If the background difference is greater than the background threshold, it means that the pixel is very different from the background at that pixel. Hence set the foreground mask to 1.

10. If the background difference is less than the background threshold, it means that the pixel is very close to the background at that pixel. Hence reset the foreground mask to 0.

Algorithm Phase III - Refining the foreground mask to reduce the effect of noise:

For all the pixels do

1. Count the number of pixels in the neighborhood of the current pixel, which are designated as foreground (that is, foreground mask is 1)
2. If a pixel designated as foreground is surrounded by enough number of foreground pixels, or a pixel designated as background is surrounded by a large number of foreground pixels, then it is treated as a foreground pixel. Set the foreground mask to 1.
3. Else set the foreground mask to 0.

In my thesis median filter is used to remove noises and undesired objects. Shyh-Yih used connected component for object classification. But this method works so hard, so more useful and flexible technique is used in this thesis. In this method, each pixel's line and column number will be sum if pixel's value is equal to '1'. Then line and column sum will be divided by sum of '1's. In conclusion absolute mass center can be calculated.

7. HUMAN MOTION ANALYSIS

Visual analysis of human motion is currently one of the most active research topics in computer vision. This strong interest is driven by a wide spectrum of promising applications in many areas such as virtual reality, smart surveillance, perceptual interface, etc. Human motion analysis concerns the detection, tracking and recognition of people, and more generally, the understanding of human behaviors, from image sequences involving humans. The emphasis is on three major issues involved in a general human motion analysis system, namely human detection, tracking and activity understanding [10].

As one of the most active research areas in computer vision, visual analysis of human motion attempts to detect, track and identify people, and more generally, to interpret human behaviors, from image sequences involving humans. Human motion analysis has attracted great interests from computer vision researchers due to its promising applications in many areas such as visual surveillance, perceptual user interface, content-based image storage and retrieval, video conferencing, athletic performance analysis, virtual reality, etc.

7.1. Application Fields

Human motion analysis has been investigated under several large research projects worldwide. For example, DARPA (Defense Advanced Research Projects Agency) funded a multi-institution project on Video Surveillance and Monitoring (VSAM), whose purpose was to develop an automatic video understanding technology that enabled a single human operator to monitor activities over complex areas such as battlefields and civilian scenes. The real-time visual surveillance system W4 employed a combination of shape analysis and tracking, and constructed the models of people's appearances to make itself capable of detecting and tracking multiple people as well as monitoring their activities even in the presence of occlusions in an outdoor environment. Researchers in the UK have also done much research on the tracking of vehicles and people and the recognition of their interactions. In addition,

companies such as IBM and Microsoft are also investing on research on human motion analysis [11].

In addition, human motion analysis shows its importance in other related areas. For instance, typical applications in virtual reality include chat-rooms, games, virtual studios, character animations, teleconferencing, etc. As far as computer games are concerned, they have been very prevalent in entertainment. Maybe people are surprised at the realism of virtual humans and simulated actions in computer games. In fact, this benefits greatly from computer graphics dealing with devising realistic models of human bodies and the synthesis of human movement based on knowledge of the acquisition of human body model, the retrieval of body pose, human behavior analysis, etc. Also, it is obvious that model-based image coding (e.g., only encoding the pose of the tracked face in images in more detail than the uninterested background in a videophone setting) will bring about very low bit-rate video compression for more effective image storage and transmission. Some examples of applications that could be realized with reliable human motion detection and tracking are:

Automated surveillance for security-conscious venues such as airports, casinos, museums, and government installations: Intelligent software could monitor security cameras and detect suspicious behavior. Furthermore, human operators could search archived video for classes of activity that they specify without requiring manual viewing of each sequence. Having automated surveillance vastly increases the productivity of the human operator and increases coverage of the surveillance.

Human interaction for mobile robotics: Autonomous mobile robots in the workplace or home could interact more seamlessly with the humans in their environment if they could reliably detect their presence. For example, robots to assist the elderly would know when assistance is needed based on the motion of a person. Safety devices for pedestrian detection on motor vehicles: Intelligent software on a camera-equipped car could detect pedestrians and warn the driver.

Automatic motion capture for film and television: Producing computer-generated imagery of realistic motion currently requires the use of a motion-capture system that stores the exact 2-D or 3-D motion of a human body using visual or radio markers attached to each limb of an actor. With accurate algorithms for human motion

tracking, the same data could be acquired from any video without any additional equipment.

The following list of applications details where most of such technology will be employed in the future:

- a) Engineering: analysis and simulation for virtual prototyping and simulation-based design.
- b) Virtual-conferencing: efficient teleconferencing using virtual representations of participants to reduce transmission bandwidth requirements.
- c) Interaction: agents and avatars that insert real-time humans into virtual worlds with virtual reality.
- d) Monitoring: acquiring, interpreting and understanding shape and motion data on human movement, performance, activities or intent.
- e) Virtual environments: living and working in a virtual place for visualization, analysis or just the experience of it. Increasingly, human dynamics is also being studied to generate virtual models of the human body using skeletal and muscular approaches.

7.2. Previous Work

The work of this thesis is an extension of the algorithms developed by Song for model-based human motion detection. Her algorithms for unsupervised learning and model detection provide the groundwork for the real-world system described herein. There are a variety of other techniques that have been developed for human motion detection. Human motion analysis has a wide range of potential applications such as smart surveillance, advanced user interface, motion based diagnosis, to name a few.

A) Visual surveillance

The strong need of smart surveillance systems stems from those security-sensitive areas such as banks, department stores, parking lots, and borders. Surveillance cameras are already prevalent in commercial establishments, while camera outputs are usually recorded in tapes or stored in video archives. These video data is currently used only “after the fact” as a forensic tool, losing its primary benefit as an active real-time media. What is needed is the real-time analysis of surveillance data

to alert security officers to a burglary in progress, or to a suspicious individual wandering around in the parking lot. Nowadays, the tracking and recognition techniques of face and gait have been strongly motivated for the purpose of access control. As well as the obvious security applications, smart surveillance has also been proposed to measure traffic flow, monitor pedestrian congestion in public spaces, compile consumer demographics in shopping malls, etc.

B) Advanced user interface

Another important application domain is advanced user interfaces in which human motion analysis is usually used to provide control and command. Generally speaking, communication among people is mainly realized by speech. Therefore, speech understanding has already been widely used in early human-machine interfaces. However, it is subject to the restrictions from environmental noise and distance. Vision is very useful to complement speech recognition and natural language understanding for more natural and intelligent communication between human and machines. That is to say, more detailed cues can be obtained by gestures, body poses, facial expressions, etc. Hence, future machines must be able to independently sense the surrounding environment, e.g., detecting human presence and interpreting human behavior. Other applications in the user interface domain include sign-language translation, gesture driven controls, and signaling in high-noise environment such as factories and airports [12].

C) Motion based diagnosis and identification

It is particularly useful to segment various body parts of human in an image, track the movement of joints over an image sequence, and recover the underlying 3-D body structure for the analysis and training of athletic performance. With the development of digital libraries, interpreting video sequences automatically using content-based indexing will save tremendous human efforts in sorting and retrieving images or video in a huge database. Traditional gait analysis aims at providing medical diagnosis and treatment support, while human gait can also be used as a new biometric feature for personal identification. Some other applications of vision-based motion analysis lie in personalized training systems for various sports, medical diagnostics of orthopedic patients, choreography of dance and ballet, etc.

Recently, some statistical methods to extract change regions from the background are inspired by the basic background subtraction methods described above. The statistical approaches use the characteristics of individual pixels or groups of pixels to construct more advanced background models, and the statistics of the backgrounds can be updated dynamically during processing. Each pixel in the current image can be classified into foreground or background by comparing the statistics of the current background model. This approach is becoming increasingly popular due to its robustness to noise, shadow, change of lighting conditions, etc.

As an example of statistical methods, Stauffer and Grimson presented an adaptive background mixture model for real-time tracking. In their work, they modeled each pixel as a mixture of Gaussians and used an online approximation to update it. The Gaussian distributions of the adaptive mixture models were then evaluated to determine the pixels most likely from a background process, which resulted in a reliable, real-time outdoor tracker which can deal with lighting changes and clutter. A recent study by Haritaoglu et al. built a statistical model by representing each pixel with three values: its minimum and maximum intensity values, and the maximum intensity difference between consecutive frames observed during the training period. The model parameters were updated periodically. The quantities which are characterized statistically are typically colors or edges. For example, McKenna et al. used an adaptive background model combining color and gradient information, in which each pixel's chromaticity was modeled using means and variances, and its gradient in the x and y directions was modeled using gradient means and magnitude variances. Background subtraction was then performed to cope with shadows and unreliable color cues effectively. Another example was Pfister, in which the subject was modeled by numerous blobs with individual color and shape statistics [8].

Temporal differencing

The approach of temporal differencing makes use of pixel-wise difference between two or three consecutive frames in an image sequence to extract moving regions. Temporal differencing is very adaptive to dynamic environments, but generally does a poor job of extracting the entire relevant feature pixels, e.g., possibly generating holes inside moving entities. As an example of this method, Lipton et al. detected moving targets in real video streams using temporal differencing. After the absolute difference between the current and the previous frame was obtained, a threshold

function was used to determine change. By using a connected component analysis, the extracted moving sections were clustered into motion regions. These regions were classified into predefined categories according to image-based properties for later tracking.

An improved version is to use three-frame differencing instead of two-frame differencing. For instance, VSAM has successfully developed a hybrid algorithm for motion segmentation by combining an adaptive

background subtraction algorithm with a three-frame differencing technique. This hybrid algorithm is very fast and surprisingly effective for detecting moving objects in image sequences.

Optical flow

Flow is generally used to describe coherent motion of points or features between image frames. Motion segmentation based on optical flow uses characteristics of flow vectors of moving objects over time to detect change regions in an image sequence. For example, Meyer et al. performed monofonic operation which computed the displacement vector field to initialize a contour-based tracking algorithm, called active rays, for the extraction of articulated objects which would be used for gait analysis.

The work by Rowley and Rehg also focused on the segmentation of optical flow fields of articulated objects. Its major contributions were to add kinematic motion constraints to each pixel, and to combine motion segmentation with estimation in EM (Expectation Maximization) computation. However the addressed motion was restricted to 2-D affine transforms. Also, in Bregler's work, each pixel was represented by its optical flow. These flow vectors were grouped into blobs having coherent motion and characterized by a mixture of multivariate Gaussians. Optical flow methods can be used to detect independently moving objects even in the presence of camera motion. However, most flow computation methods are computationally complex and very sensitive to noise, and cannot be applied to video streams in real-time without specialized hardware. More detailed discussion of optical flow can be found in Barron's work [11].

Object classification

Different moving regions may correspond to different moving targets in natural scenes. For instance, the image sequences captured by surveillance cameras mounted in road traffic scenes probably include pedestrians, vehicles, and other moving objects such as flying birds, flowing clouds, etc. To further track people and analyze their activities, it is very necessary to correctly distinguish them from other moving objects. The purpose of moving object classification is to precisely extract the region corresponding to people from all moving blobs obtained by the motion segmentation methods discussed above. Note that this step may not be needed under some situations where the moving objects are known to be human. For the purpose of describing the overall process of human detection, we present a simple discussion on moving object classification here. At present, there are two main categories of approaches towards moving object classification.

Recently, Kojima et al. proposed a new method to generate natural language description of human behaviors appearing in real video sequence. First, a head region of a human, as a portion of the whole body, was extracted from each image frame, and its three-dimensional pose and position were estimated using a model-based approach. Next, the trajectory of these parameters was divided into the segments of monotonous movement. The conceptual features for each segment, such as degrees of change of pose and position and that of relative distance from other objects in the surroundings, were evaluated. Meanwhile, the most suitable verbs and other syntactic elements were selected. Finally, the natural language text for interpreting human behaviors was generated by machine translation technology.

8. CONCLUSIONS AND RECOMMENDATIONS

Human always wants to live in safe environments. People have been searching different methods to supply their security since starting of human life in covered area. The development of these methods which improved to reach this target, depends on the development of technology. In the past, people security was obtained by human and animals. But today, security concept is called to mind with machines. The main topic of security systems is tracking the moving objects. Tracking the moving object (especially human) has great importance for these systems.

In this thesis, it is aimed to improve an alternative security system model with using image processing methods. In this model, movement of object is detected and object position is marked on area's map. In this study, using elements of image processing enabled us to calculate the position of moving object in a room. These data can be useful in many security systems such as home security systems, prison security systems, etc. We had to stand in requirements of scalability and compatibility in order to enable coping with a changes in the environment's conditions and the camera's location. So we developed the system modular approach by using a number of modules and Object Oriented Programming (OOP). The software was developed for intermediate level users. That's in order to ease changes executions and supporting compatibility to a future requirements. In order to enable changes in the location of the cameras and the cameras parameters influence on the image processing, A Graphical User Interface (GUI) was built which enable easily changing of the parameters during the program execution. Thus the optimal parameters could be investigated and found dynamically.

The algorithm presented in this thesis was designed to be efficient enough for real-time applications. For video running at 30 frames per second with a resolution of 640 pixel by 480 pixel, real-time performance would require each frame to be processed in 33.3 milliseconds or less. Background subtraction 1.0 ms per frame, Finding features 1.8 ms per object, position computations and filtering 12.5 ms per frame, tracking features 2.5 ms per object. These results demonstrate that for a frame with

one object, the total processing time is about 20 ms. The system presented herein is successful as a robust detector of human motion. However, there is still much progress to be made, We make the fundamental assumption that the background will dominate most scenes and will be stationary. This is unacceptable for more crowded environments where surveillance is desired. In addition, the requirement for a stationary background rules out situations where the camera is in motion such as on vehicles or robotic cameras. Futhermore, any foreground objects that overlap or are close to each other will be treated as a single object. This undesired grouping is the nature of the connected component algorithm currently used by the contour finder. In order to successfully handle these cases, a better algorithm for image segmentation is needed. One that relies on optical flow rather background subtraction would be more robust in these situations. However, such an algorithm would be far more expensive in terms of runtime performance.

The system presented herein is successful as a robust detector of human motion. However, there is still much progress to be made, We make the fundamental assumption that the background will dominate most scenes and will be stationary. This is unacceptable for more crowded environments where surveillance is desired. In addition, the requirement for a stationary background rules out situations where the camera is in motion such as on vehicles or robotic cameras. Futhermore, any foreground objects that overlap or are close to each other will be treated as a single object. This undesired grouping is the nature of the connected component algorithm currently used by the contour finder. In order to successfully handle these cases, a better algorithm for image segmentation is needed. One that relies on optical flow rather background subtraction would be more robust in these situations. However, such an algorithm would be far more expensive in terms of runtime performance. There are a variety of enhancements that could be made to this system to achieve greater detection accuracy and increased robustness: Objects could be tracked between frames rather than simply performing human motion detection on single frames. For example, a Kalman filter could be used to predict the future position and human likelihood of a given object. Such a filter would smooth out particular frames in which detection fails, and would eliminate many false detections. The net effect would be an improvement in both PD and PFA along with a smooth tracking capability useful for higher-level applications. As described above, the current model

of motion does not take into account the time-dependent nature of a dynamic background. Much greater accuracy would be possible with a detector and model that takes advantage of this periodicity in time. The current background subtraction algorithm can be confused by fast lighting changes or moving shadows. A better algorithm would use a technique based on optical flow for the image segmentation. This approach would also allow the camera to be in motion relative to the background.

Fast and accurate motion segmentation is a significant but difficult problem. The captured images in dynamic environments are often affected by many factors such as weather, lighting, clutter, shadow, occlusion, and even camera motion. Taking only shadow for an example, they may either be in contact with the detected object, or disconnected from it. In the first case, the shadow distorts the object shape, making the use of subsequent shape recognition methods less reliable. In the second case, the shadow may be classified as a totally erroneous object in the natural scenes. Nearly every system within human motion analysis starts with segmentation, so segmentation is of fundamental importance. Although current motion segmentation methods mainly focus on background subtraction, how to develop more reliable background models adaptive to dynamic changes in complex environments is still a challenge.

REFERENCES

- [1] **P. Salembier and F. Marques**, 1999. Region-based representations of image and video: Segmentation tools for multimedia services. *IEEE Transactions on Circuits and Systems for Video Technology*.
- [2] **O. Veksler**, 2000. Image segmentation by nested cuts. *IEEE Computer Vision and Pattern Recognition*.
- [3] **Andrews H.C.**, 1998. Computer Techniques In Image Processing, *Academic Press*.
- [4] **Aggarwal J.K.**, 1997. Computer Methods In Image Analysis, *IEEE Press*.
- [5] **Ballard D.H.**, 1982. Computer Vision, Prentice-Hall.
- [6] **Stoffel J.C.**, 1982. Graphical And Binary Image Processing , Artech House.
- [7] **Hall E.**, 1979. Computer Image Processing And Recognition, *Academic Press*.
- [8] **A. Brandt E. Sharon and R. Basri**, 2000. Fast multiscale image segmentation. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.
- [9] **J. Malik and J.**, 1998. Shi. Motion segmentation and tracking using normalized cuts. *International Conf. on Computer Vision*.
- [10] **S. Y. Ma S. Y. Chien and L. G. Chen**, 2000. Efficient moving object recognition algorithm using background registration technique. *IEEE Transactions on Circuits and Systems for Video Technology*.
- [11] **Z. Wu and R. Leahy**, 1993. An optimal graph theoretic algorithm to data clustering: Theory and its application to image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- [12] **Shao-Yi Chien, Shyh-Yih Ma, and Liang-Gee Chen**, 2002. Efficient moving object segmentation algorithm using background registration technique, *IEEE Transactions on Circuits and Systems for Video Technology*.

AUTOBIOGRAPHY

Ferhat UÇAN was born in 1979 in Istanbul. He graduated from Adile Mermerci Anatolian High School in 1997. He completed his undergraduate study in Computer Engineering Department at Yıldız Technical University in 2002. He attended to his graduate study in the programme of Computer Engineering at Istanbul Technical University in 2002. He has been working as a researcher in Tubitak Marmara Research Center Information Technologies Research Institute since April 2003.

