

Lokesh Peddireddi

# Object Tracking and Velocity Determination using TMS320C6416T DSK

Internship Project Report

Studium Information Technology

---

Alpen-Adria-Universität Klagenfurt  
Fakultät für Technische Wissenschaften



Begutachter: Univ.–Prof. Dr. techn. Bernhard Rinner  
Betreuer: Univ.–Ass. Dipl.–Ing. Wolfgang Schriebl

Institute of Networked and Embedded Systems  
Pervasive Computing

Klagenfurt, im April 2008

## **Abstract**

This project deals with the tracking and following of single object in a sequence of frames and the velocity of the object is determined. Algorithms are developed for improving the image quality, segmentation, feature extraction and for determining the velocity. The developed algorithms are implemented and evaluated on TMS320C6416T DSP Starter Kit (DSK). Segmentation is performed to detect the object after reducing the noise from that scene. The object is tracked by plotting a rectangular bounding box around it in each frame. The velocity of the object is determined by calculating the distance that the object moved in a sequence of frames with respect to the frame rate that the video is recorded. The algorithms developed can also be used for other applications (real time, object classification, etc.).

## Acknowledgments

I would like to thank my project advisor Mr. Wolfgang Schriebl, Pervasive Computing Group, for constantly supporting me throughout the project and guiding me in a proper way.

I am grateful to Prof. Dr. techn. Bernhard Rinner, Pervasive Computing Group, Alpen-Adria University, for giving me a very good opportunity to carry out my internship work in this group and for providing me sufficient resources to accomplish this project.

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Motivation . . . . .	1
1.2	Goal . . . . .	1
1.3	Outline of the Project . . . . .	1
<b>2</b>	<b>Single Object Tracking</b>	<b>2</b>
2.1	Pre-processing . . . . .	2
2.1.1	Mean Filter . . . . .	2
2.1.2	Gaussian Smoothing . . . . .	3
2.1.3	Median Filter . . . . .	3
2.2	Segmentation . . . . .	3
2.2.1	Histogram Based Segmentation . . . . .	3
2.2.2	Single Gaussian Background Method . . . . .	4
2.2.3	Frame Difference . . . . .	4
2.3	Feature Extraction . . . . .	4
2.3.1	Edges . . . . .	5
2.3.2	Bounding Box with Colour Feature . . . . .	5
2.4	Object Detection . . . . .	5
2.4.1	Optical Flow . . . . .	5
2.4.2	Block Matching . . . . .	6
2.5	Tracking . . . . .	6
2.5.1	Velocity . . . . .	6
<b>3</b>	<b>Algorithms</b>	<b>7</b>
3.1	Noise removal . . . . .	7
3.2	Segmentation . . . . .	8
3.3	Feature Extraction . . . . .	8
3.4	Distance . . . . .	9
3.5	Velocity . . . . .	9
<b>4</b>	<b>Implementation and Evaluation</b>	<b>10</b>
4.1	Hardware Specification . . . . .	11
4.2	Conversion . . . . .	11
4.2.1	Video to images . . . . .	11
4.2.2	Image to hexadecimal format . . . . .	12
4.3	Problems . . . . .	12

4.4	Noise Removal . . . . .	13
4.5	Segmentation . . . . .	13
4.6	Feature Extraction . . . . .	14
4.7	Distance . . . . .	14
4.8	Velocity . . . . .	15
<b>5</b>	<b>Results</b>	<b>17</b>
<b>6</b>	<b>Conclusion</b>	<b>19</b>
	<b>Bibliography</b>	<b>20</b>

# List of Figures

2.1	Single gaussian background method . . . . .	4
2.2	frame difference between 2 frames . . . . .	5
4.1	Block Diagram of Evaluation and implementation . . . . .	10
4.2	Frame difference . . . . .	14
4.3	Bounding box with centroid . . . . .	15
4.4	Distance and velocity of moving object . . . . .	16
5.1	Step by step diagrams during evaluation and implementation . . . . .	18

# Chapter 1

## Introduction

This project deals with the single object tracking and velocity of that object. The evaluation is performed on TMS320C6416T DSP starter Kit (DSK).

### 1.1 Motivation

The motivation behind this project is to develop software for tracking, the major application in security, surveillance and vision analysis. The developed software must be capable of tracking any single object moving in the frame and to implement on a hardware which is capable of on board calculations with high performance and low power consumption. This system might be useful for extending in real-time surveillance or object classification.

### 1.2 Goal

Goal of this project is to develop an algorithm for tracking of an object and determining the velocity of moving object in sequence of frames. All the evaluation has to be performed on TMS320C6416T DSP starter Kit (DSK), which is capable of doing onboard calculations. It has features like internal memory and external memory on board to store instructions and doing evaluation on board. It has special functions for handling graphics up to ten-times faster than others. Algorithms can be extended for real time applications.

### 1.3 Outline of the Project

The implementation of object tracking and its velocity determination is explained step by step in this report. In chapter 2, few approaches are implemented in real-time for object tracking and velocity determination are explained. In chapter 3, algorithms that are developed to accomplish this project are explained. In chapter 4, implementation part which includes the logical approach of tracking done for the project is described. In chapter 5, evaluation part describes the interface between the hardware and workstation, converting the video into images and loading the images. In chapter 6, results are visualized and finally concluded.

## Chapter 2

# Single Object Tracking

Object tracking is the process of locating and following the moving object in sequence of video frames. Smart cameras are used as input sensors to record the video. The recorded video may have some noise due to bad weather (light, wind, etc. or due to problems in sensors). Few algorithms are tested to improve the image quality, to detect moving object, calculation of distance and velocity of the moving object.

### 2.1 Pre-processing

The pre-processing performs some steps to improve the image quality. Few algorithms are explained and filtering is done by using box-filter techniques.

**Box functionality** : If we take an image with X\*Y resolution, the pixels starting from 2nd row 2nd column to X-1 row Y-1 column are scanned with all the neighbouring elements, they form a shape like box. The pixels from 2nd row 2nd column to X-1 row Y-1 column are taken into consideration since we need to scan all the surrounding elements and for the 1st row ,1st column, last row and last column elements are not possible to scan there surrounding elements as they don't have.

#### 2.1.1 Mean Filter

Mean filter is the simplest type of low pass filter to remove noise. The noise is removed using the box functionality, which scans the whole image. For every pixel the mean from the box elements are calculated and the pixel values are stored in to the central element. Let us consider an example with 3x3 matrixes

$$\begin{bmatrix} 1 & 2 & 9 \\ 4 & \mathbf{3} & 8 \\ 5 & 6 & 7 \end{bmatrix}$$

$$Pixel.value = \frac{1}{9} \cdot (1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9)$$

in the above matrix, central element is '3' and after calculating the pixel value, the value of



3 is replaced by the pixel value. The calculation time for mean filter is very less compared to all other. By using this filter smoothing is done. [11] [8]

### 2.1.2 Gaussian Smoothing

Gaussian smoothing operator is used to blur images and remove noise. Gaussian smoothing has the similarity of mean filter, but uses a different function to calculate the pixel value. A box is scanned over the whole image and the pixel value calculated from the standard deviation of Gaussian is stored in the central element. The size of the box may vary from 1 to 7 means 1x1 to 7x7 elements. [10]

$$G(x, y) = \frac{1}{2\pi\sigma^2} \cdot e^{-\left(\frac{x^2 + y^2}{2 \cdot \sigma^2}\right)}$$

Where x and y denotes the positions of elements in the box.

### 2.1.3 Median Filter

The median filter is a classical noise removal filter. Noise is removed by calculating the median from all its box elements and stores the value to the central element. If we consider and example of 3x3 matrix

$$\begin{bmatrix} 1 & 2 & 9 \\ 4 & \mathbf{3} & 8 \\ 5 & 6 & 7 \end{bmatrix}$$

The median filter sorts the elements in a given matrix and median value is assigned to the central pixel. Sorted elements 1, 2, 3, 4, 5, 6, 7, 8, 9 and median 5 will assign to the central element. Similar box scan is performed over the whole image and reduces noise. Execution time is more compared to mean filter, since the algorithm involves with sorting techniques. But it removes the small pixel noise noise.[9][6]

## 2.2 Segmentation

Segmentation is the process of dividing digital image into multiple regions. Segmentation shows the objects and boundaries in an image. Each Pixel in the region has some similar characteristics like colour, intensity, etc. Few methods for segmenting the images are explained below.

### 2.2.1 Histogram Based Segmentation

One of the simple ways of doing segmentation is using histogram. Histogram is computed for all the image pixels. The peaks in histogram are produced by the intensity values that are produced after applying the threshold and clustering. The pixel value is used to locate the regions in the image. Based on histogram values and threshold we can classify the low intensity values as object and the high values are background image (most of the cases). [12]

### 2.2.2 Single Gaussian Background Method

Single gaussian background model is used to separate the background and foreground objects. It is a statically method of separation. In this a set of frames (previous frames) are taken and the calculation is done for separation. The separation is performed by calculating the mean and variance at each pixel position. If we take  $\mathbf{N}$  frames with pixel value  $\mathbf{P}$  and intensity  $\mathbf{I}$ .

$$\mu = \frac{1}{p} \cdot \sum_{i=1}^p [I(i)]$$

$$\sigma = \frac{1}{p} \cdot \sum_{i=1}^p [I(i) - \mu]$$

Now after calculating the variance of each pixel, a threshold function is used to separate the foreground and background objects. Figure 2.1 shows the single gaussian background method while testing.[13], [9]

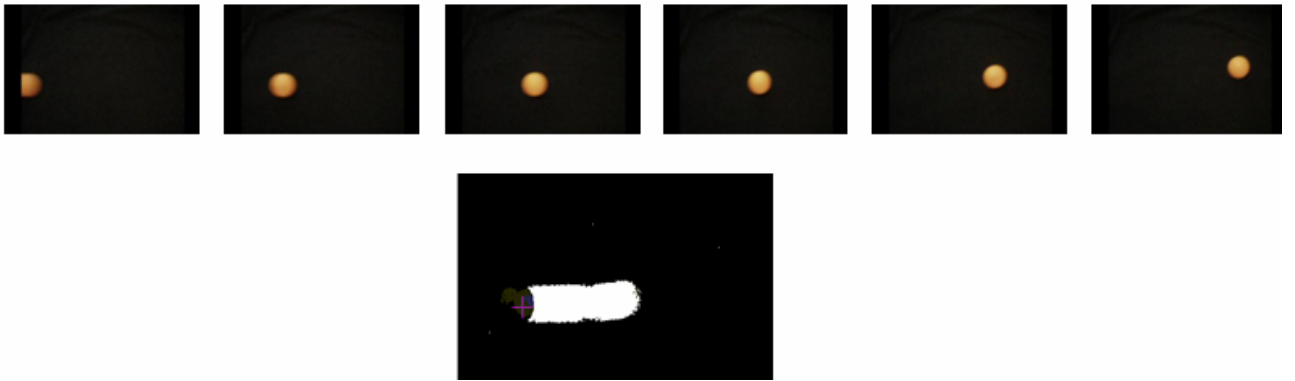


Figure 2.1: Single gaussian background method

### 2.2.3 Frame Difference

Frame difference calculates the difference between 2 frames at every pixel position and store the absolute difference. It is used to visualize the moving objects in a sequence of frames. It takes very less memory for performing the calculation.[13]

Let us consider an example, if we take a sequence of frames, the present frame and the next frame are taken into consideration at every calculation and the frames are shifted (after calculation the next frame becomes present frame and the frame that comes in sequence becomes next frame). Figure 2.2 shows the frame difference between 2 frames.

## 2.3 Feature Extraction

Feature Extraction plays a major role to detect the moving objects in sequence of frames. Every object has a specific feature like colour or shape. In a sequence of frames, any one of the feature is used to detect the objects in the frame.

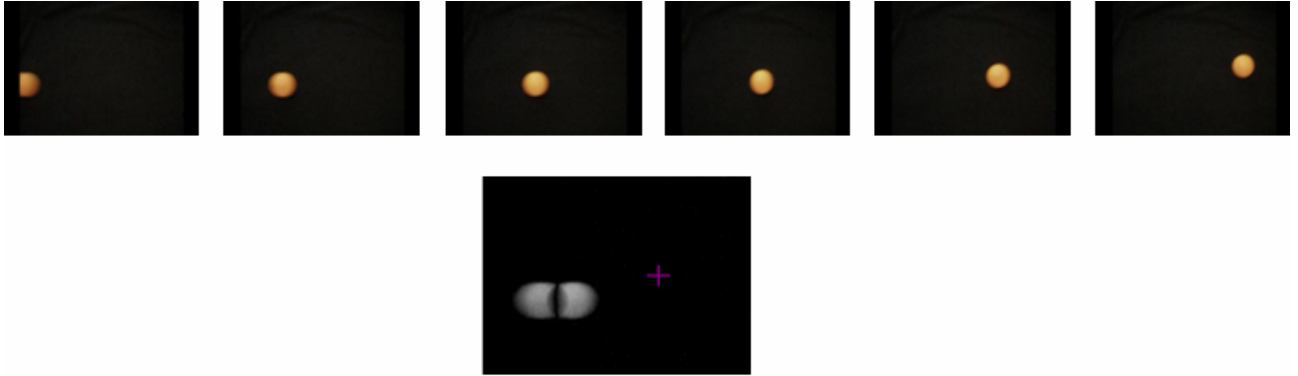


Figure 2.2: frame difference between 2 frames

### 2.3.1 Edges

Edges are formed where there is a sharp change in the intensity of images. If there is an object, the pixel positions of the object boundary are stored and in the next sequence of frames this position is verified. Corner based algorithm uses the pixel position of edges for defining and tracking of objects .[14]

### 2.3.2 Bounding Box with Colour Feature

If the segmentation is performed using frame difference, the residual image is visualized with rectangular bounding box with the dimensions of the object produced from residual image. For a given image, a scan is performed where the intensity values of the image is more then limit (depends on the assigned value, for accurate assign maximum). In this Features is extracted by colour and here the intensity value describes the colour. The pixel values from the first hit of the intensity values from top, bottom, left and right are stored. By using this dimension values a rectangular bounding box is plotted within the limits of the values produced.

## 2.4 Object Detection

Extraction of objects using the features is known as object detection. Every object has a specific feature based on its dimensions. Applying feature extraction algorithm, the object in each frame can be pointed out.

### 2.4.1 Optical Flow

Optical flow is one way to detect moving objects in a sequence of frames. In this, the vector position of pixels is calculated and compared in sequence of frames for the pixel position. Typically the motion is represented as vector position of pixels. [7]

### 2.4.2 Block Matching

Block matching algorithm is a standard technique for determining the moving object in video. Blocks are formed in a region without overlapping on the other region. Every block in a frame is compared to the corresponding blocks in the sequence of frames and compares the smallest distance of pixel values. [5]

## 2.5 Tracking

The process of locating the moving object in sequence of frames is known as tracking. This tracking can be performed by using the feature extraction of objects and detecting the objects in sequence of frames. By using the position values of object in every frame, we can calculate the position and velocity of the moving object.[13]

### 2.5.1 Velocity

The velocity of moving object is calculated by the distance it travelled with respect to the time. Euclidean distance formula is used to calculate the distance between the sequences of frames. By using the values of distance with respect to frame rate, the velocity of the object is defined. The defined velocity is of 2-dimension (since camera is static).

## Chapter 3

# Algorithms

This chapter explains the algorithms to track the single object and to estimate the velocity of moving object. The sequential approach to track the moving objects and the velocity of objects are as follow.

- Noise removal: To improve the image quality
- Segmentation: To separate multiple regions in image
- Feature Extraction: To analyze the regions in image
- Tracking : Analyzing the position, velocity and moving direction of object

### 3.1 Noise removal

A noise removal algorithm is developed by using the median filter as explained in section 2.1.3. Algorithm for noise removal is explained as follow

1. Read the input image
2. for (present position=initial position: final position)
  - (a) Scan all the surrounding elements
  - (b) Use bubble sort technique to sort all the values
  - (c) Calculate the median of it
3. Assign the value to the 'present position' of the input image

initial position = 2nd row, 2nd column of an image resolution

final position = (n-1) row, (n-1) column of an image resolution

present position = the pixel value and it varies from initial to final position

Values are taken from 2nd row-2nd column to (n-1) row-(n-1) column because we need to scan all the surrounding elements of each pixel.

## 3.2 Segmentation

To perform the segmentation operation, frame difference algorithm is implemented as it takes less processing time. Frame difference algorithm performs separation of two sequential frames and it is explained in detail in section 2.2.3.[13] Algorithm for the segmentation is explained as follow

1. Read the input images
2. for (present position=initial position: final position)
  - (a) Difference between the pixels values at present position of two images is calculated
  - (b) Calculate the absolute value
  - (c) Store the difference in new image at same pixel position that is at present position

## 3.3 Feature Extraction

Every object has a specific feature which is used to visualize the object and used for tracking. After performing the segmentation, a rectangular bounding box is plotted with the dimensions of the object produced in the residual image. Section 2.3.2 explains a clear view on bounding box. Algorithm for the bounding box is as followed

1. Read the image difference
2. for (present position=initial value: final value) of Y resolution
3. for (present position=initial value: final value) of X resolution
  - (a) calculate the sharp change in intensity of image from top and bottom
  - (b) store the values in an array
4. Height of the bounding box is = bottom value- top value
5. for (present position=initial value: final value) of X resolution
6. for (present position=initial value: final value) of Y resolution
  - (a) calculate the sharp change in intensity of image from left and right
  - (b) store the values in an array
7. Width of the bound box = right value - left value
8. Using the dimensions, draw boundary to the image  
 Initial value: the starting position of the pixel in an image. Final value: the ending position of the pixel in an image.

$$Height = \frac{bottom.value - top.value}{2}$$

$$Width = \frac{right.value - left.value}{2}$$

9. Add the height value with the top value and store it in a variable like mid top
10. Add the width value to the left value and store it in a variable like mid left
11. Assign the max intensity to the pixel at pixel value at (mid top, mid left)

### 3.4 Distance

The distance travelled by the object is determined by using the centroid. It is calculated by using the Euclidean distance formula. The variables for this are the pixel positions of the moving object at initial stage to the final stage. Algorithm for calculating distance is explained as follow

1. Read the centroid position of each image.
2. Calculate the distance between two centroid images
3. for (present position=initial value: final value) of X resolution
4. for (present position=initial value: final value) of Y resolution
5. Calculate change in distance by

$$Distance = \sqrt{(X2 - X1)^2 + (Y2 - Y1)^2}$$

Where X1=previous pixel position and X2=present pixel position in width  
Y1=previous pixel position and Y2=present pixel position in height

6. Store all the distance values in an Array.

### 3.5 Velocity

Velocity of moving object is determined using the distance travelled by the centroid to the frame rate of the video. Section 2.5.1 explains the velocity in 2-dimension. Algorithm for calculating velocity is explained as follow

1. Read the distance travelled by the object

$$Velocity = \frac{distance.travelled}{frame.rate}$$

3. Save the value in an array
4. The velocity of moving object in the sequence frames is defined in pixels / second

## Chapter 4

# Implementation and Evaluation

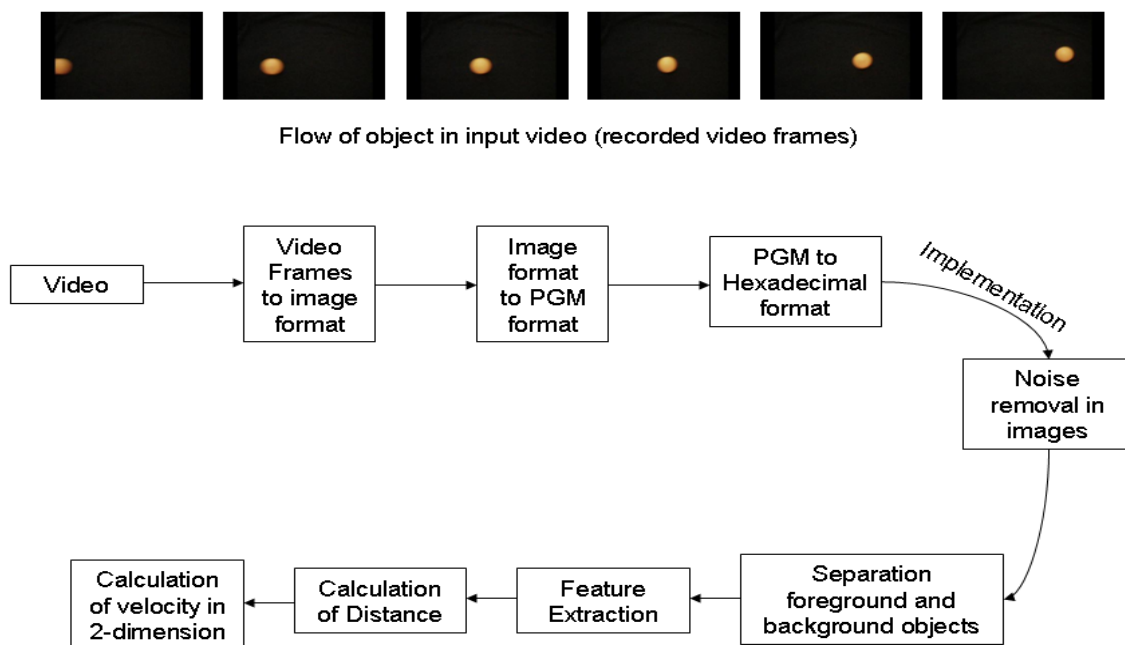


Figure 4.1: Block Diagram of Evaluation and implementation

This chapter explains the implementation and evaluation of algorithms for single object tracking. The step by step process is shown graphically in figure 4.1. Algorithms for the single object tracking are developed in C language and evaluated on TMS320C6416T DSP Starter Kit (DSK). The task is performed by recording a video using digital camera with 30fps. The implementation is initially performed on matlab and various methods for object tracking are tested. Among the tested methods, frame difference is used for further



calculations, since it satisfies the available conditions (execution time and memory) of hardware. The initial steps before object tracking include video to image conversion, images to input format of the TMS320C6416T. The algorithms that are created are deployed into TMS320C6416T DSK by using the code composer studio.

## 4.1 Hardware Specification

TMS320C6416T DSK is a low-cost standalone development platform that enables users to evaluate and develop applications for C64XX DSP family. It is the Highest-Performance Fixed-Point Digital Signal Processors (DSPs) with Advanced Very-Long-Instruction-Word (VLIW). The DSK also serves as a hardware reference design for the TMS320C6416T DSP. The DSP on the 6416T DSK interfaces to on-board peripherals through one of two busses; there are 2 external memories with 64-bit wide External Memory Interface Function A (EMIFA) and the 8-bit wide External Memory Interface Function B (EMIFB). The SDRAM, Flash and CPLD are each connected to one of the busses. EMIFA is also connected to the daughter card expansion connectors which are used for third party add-in boards. Each EMIF (External Memory Interface) has 4 separate addressable regions called chip enable spaces (CE0-CE3). The SDRAM occupies CE0 of EMIFA while the CPLD and Flash are mapped to CE0 and CE1 of EMIFB respectively. Daughter cards use CE2 and CE3 of EMIFA. It is operated at 1 Gigahertz frequency and has 16 Mbytes of synchronous DRAM and 512 Kbytes of non-volatile Flash memory.

Code composer studio software is used as interface between the TMS320C6416T DSP starter kit (DSK) and the workstation (computer). The input data must be in hexadecimal format for the code composer studio. Code Composer communicates with the DSK through an embedded Joint Test Action Group (JTAG) emulator with a USB host interface. The DSK can also be used with an external emulator through the external JTAG connector. [4]

From the available hardware memory, SDRAM memory is used for storing the input images, temporary images and the final output images. The starting address of the SDRAM is from 0x80000000 and the available space is till 0x90000000.

## 4.2 Conversion

Conversion explains the functions that are performed to convert the recorded video into input format of image to code composer studio. The steps for the conversion are

- Video to images
- Images to hexadecimal format

### 4.2.1 Video to images

The task of object tracking is performed by a video recorded with digital camera. The recorded video is in format of **.mov** with 30 fps. The video is converted into **.avi** format with 20 fps using ULEAD VIDEO STUDIO 10.0 [3]. The frame rate of the video is converted to

- reduce the frame size of video clip as we can see all the actions in less frames
- less memory can be used for calculations
- change in position of object can be easily observed in each frame

The converted video frames is now spitted into individual frames and converted each into **jpg** image format using Image Video Machine 3.2 software.[2].

#### 4.2.2 Image to hexadecimal format

The images are converted into hexadecimal format by using a simple C language program. This program takes **pgm** format images as input and converts it into hexadecimal. The initial condition, we convert the available images into **pgm** format, it is done by using GIMP software [1] which converts **jpg** format to **pgm** format). After converting the image to **pgm** format, C program is executed for each image and exported to hexadecimal format of the image. Algorithm of the C program used for conversion is explained as follow

1. Read the input image
2. Create a temporary array to store the image data
3. Create a file to store the output image
4. while (end of pixel values)
  - (a) Convert the pixel value into hexadecimal value and store them in temporary created array
  - (b) for (every 4 pixel values)
    - Store the hexadecimal values in the output file in reverse order.
    - Shift to new line

In this, hexadecimal values are stored in row and is shifted to new line after 4 bits. The output image is imported into code composer studio by assigning the starting address of it. This starting address is used as input reference for the implementation part.

Ex: The starting address of the input image is 0x80000000 (Address I used in implementation) in code composer studio.

### 4.3 Problems

The problems that are faced during the evaluation part are

- *Memory* : Out of memory error, during execution. Due to overlap of image memory slots with instructions memory. This problem is reduced by changing the memory location (changed from ISRAM to SDRAM).
- *Endian Problem* : The problem depending on the hardware and operating system. The image pixel order changes to reverse order due to this endian. This can be reduced by storing the hexadecimal values in the reverse order in each line.

## 4.4 Noise Removal

Noise removal is used to improve the image quality by reducing noise. Noise is reduced by filter technique and by adjusting the threshold value. Median filter algorithm is applied as it removes the salt pepper noise. The median filter is very good at removing short noise (but not perfect). Section 2.1.3 explains the overview of median filter and section 3.1 explains the algorithm for the median filter. When the algorithm is implemented, a temporary memory is allotted for the temporary image and the values of the median filter are stored in the temporary image. The address of the temporary image is passed and used as input address for the segmentation.

Ex (address used in implementation):

- 0x8000000, the starting address of the input image to the median filter.
- 0x80020000, the address created for temporary storage of image after median filter.

When the implementation is done, 2 temporary memory allocations are created for 2 temporary images (for sequence of images (present and previous)). At the end of each loop (after calculating the distance and velocity) the temporary image in the present image is stored in the previous image and the when the filter operation is done to the next image, it is stored in present image. Ex (address used in implementation): Temporary memory for temporary images is 0x80020000 0x80021000 (for present and previous images)

## 4.5 Segmentation

Segmentation is the process of analyzing multiple regions in a frame. Frame difference algorithm is used to separate the moving objects in frame. Advantage of using frame difference over the others is less memory usage and two frames are needed for calculation and gives good results when single moving object is to be tracked. Section 2.2.3 gives an overview about frame difference. The algorithm for the calculation is explained in section 3.2 explains the algorithm part of frame difference. When the algorithm is implemented, the starting address of the temporary images (both present and previous images) is taken and the image after the frame difference is stored in the memory allocated for the output image. Figure 4.2 shows the segmentation of object which is the residual image of frame difference. [13]

Ex (address used in implementation) :

- 0x80020000 0x80021000, the starting address of the images from noise removal are used as inputs for frame difference
- Output from frame difference is stored at 0x80010000 (initially allocated address for output image).

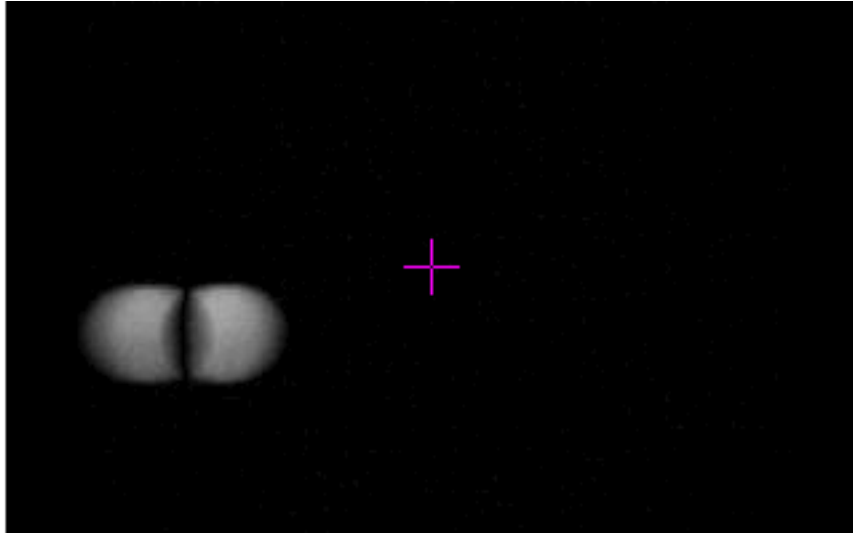


Figure 4.2: Frame difference

## 4.6 Feature Extraction

Feature Extraction visualizes the moving object in sequence of frames and a rectangle bounding box is drawn using its dimensions. Section 2.3.2 explains a clear view on bounding box. Bounding box algorithm described in 3.3 plots the bounding box to the residual image from the frame difference. When the algorithm is implemented for the image, dimensions of the image are produced and a rectangular boundary is plotted. By using this bounding box dimensions, we plot the centroid of the box and used for tracking of objects and determining the velocity of object. Figure 4.3 shows the bounding box for the residual image produced from the segmentation.

Ex (address used in implementation):

- 0x80010000 is the input for bounding box
- Dimensions are scanned for the object in image and a bounding box is plotted on the same image
- A centroid is plotted from the dimensions of the bounding box

## 4.7 Distance

The distance travelled is defined as the distance travelled by the centroid pixel in each image. In this 2 arrays are created to store all the pixel position values of the centroid in every frame and to store the distance between previous and present pixel positions. Section 3.4 explains the algorithm of calculating the distance. When the algorithm is implemented then the pixel positions are stored in the array and the distance is calculated from previous and present position by using Euclidean distance formula and stored in the array created

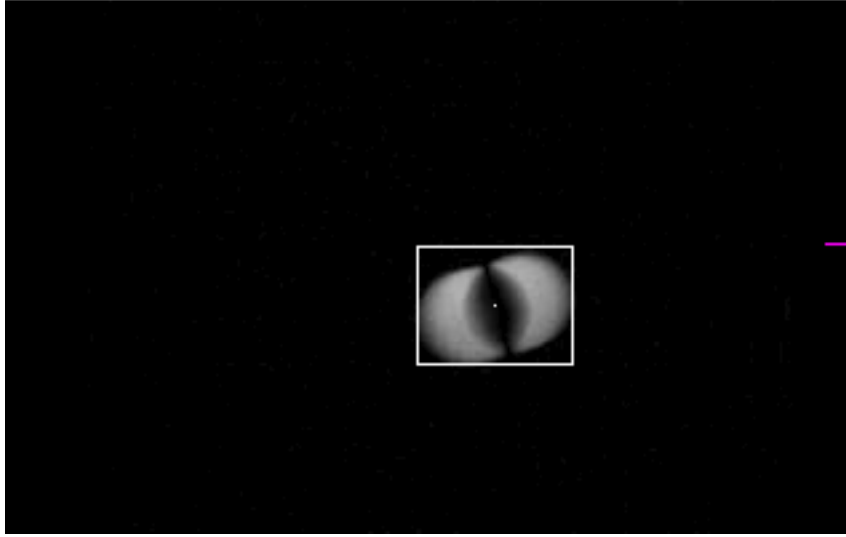


Figure 4.3: Bounding box with centroid

for the distance.

Ex (while implementation):

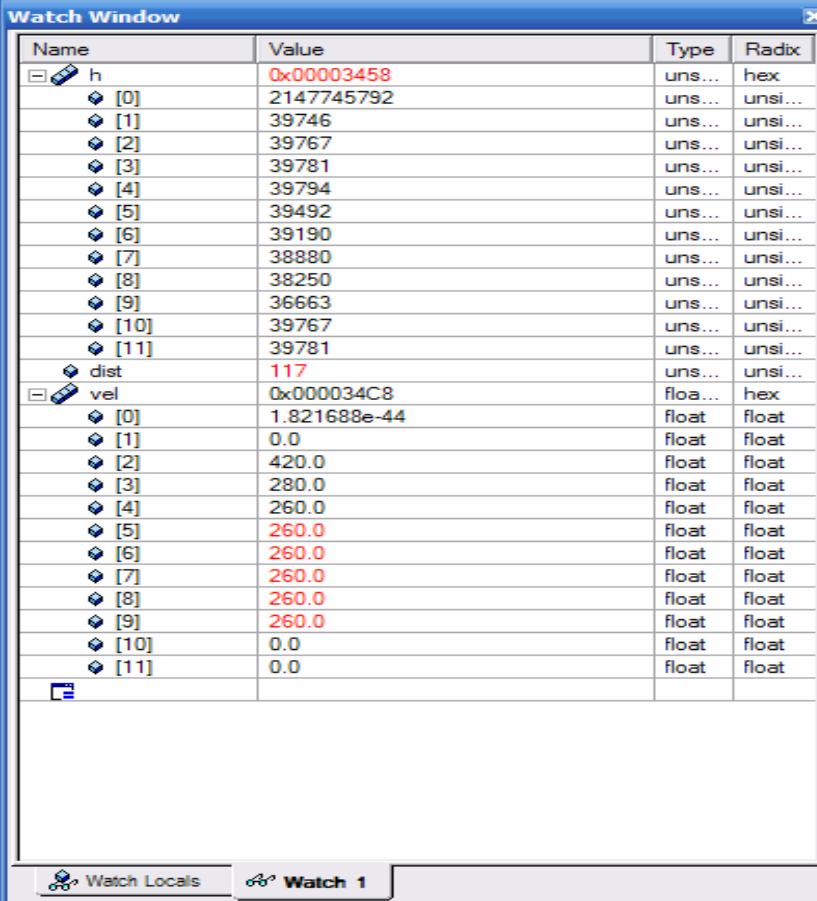
- 'h' an array created for pixel position in each frame.

## 4.8 Velocity

Velocity of moving object is determined by the distance travelled by the centroid in each frame to the frame rate. Section 2.5.1 explains how to calculate the velocity. Section 3.5 explains the algorithm for velocity calculation in 2-dimension (because of static camera). When the algorithm is implemented an array is created to store the velocity values. After calculating the velocity, the values are stored in the array created. Figure 4.4 shows the sample values of velocity and distance in table. The array denoted by "h" in the figure shows the pixel position of centroid in sequence of frames. The array denoted by "vel" shows the velocity of centroid in a set of sequence of frames.

Ex (while implementation):

- array with name h shows the values of the pixel position
- array with name vel shows the values of the velocity it moved in 2-dimension direction



The screenshot shows a 'Watch Window' with a table of variables and their values. The table has four columns: Name, Value, Type, and Radix. The variables are grouped into three sections: 'h' (array of 12 unsigned integers), 'dist' (unsigned integer), and 'vel' (array of 12 floats). The values for 'h' range from 2147745792 to 39781. The value for 'dist' is 117. The values for 'vel' range from 1.821688e-44 to 0.0. The 'Type' column shows 'uns...' for unsigned integers and 'float' for floats. The 'Radix' column shows 'hex' for hexadecimal and 'float' for floating-point. The 'Watch 1' tab is selected at the bottom.

Name	Value	Type	Radix
h	0x00003458	uns...	hex
[0]	2147745792	uns...	unsi...
[1]	39746	uns...	unsi...
[2]	39767	uns...	unsi...
[3]	39781	uns...	unsi...
[4]	39794	uns...	unsi...
[5]	39492	uns...	unsi...
[6]	39190	uns...	unsi...
[7]	38880	uns...	unsi...
[8]	38250	uns...	unsi...
[9]	36663	uns...	unsi...
[10]	39767	uns...	unsi...
[11]	39781	uns...	unsi...
dist	117	uns...	unsi...
vel	0x000034C8	fla...	hex
[0]	1.821688e-44	float	float
[1]	0.0	float	float
[2]	420.0	float	float
[3]	280.0	float	float
[4]	260.0	float	float
[5]	260.0	float	float
[6]	260.0	float	float
[7]	260.0	float	float
[8]	260.0	float	float
[9]	260.0	float	float
[10]	0.0	float	float
[11]	0.0	float	float

Figure 4.4: Distance and velocity of moving object

## Chapter 5

# Results

Object tracking, the main application for security, surveillance and vision analysis, is evaluated on TMS320C6416T DSK by using the code composer studio, the interface to deploy the algorithms from the workstation to TMS320C6416T. In this, a video is recorded using digital camera. The recorded video frames are converted into individual frame and then converted into Portable Gray Map (PGM) format images. C language code is implemented to convert the pixel values in the PGM format image into Hexadecimal values and store the values in the files, which are inputs for code composer studio. The implementation part starts when the images in Hexadecimal format are imported into code composer studio. In the implementation part, noise is removed for the imported images using median filter. The filtered images are used as input for the frame difference for the separation of foreground and background objects. A rectangular bounding box is plotted around the foreground objects produced from frame difference. By using the dimensions of rectangular bounding box, a centroid is plotted. The position of the centroid is stored in the array and the distance is calculated using Eucliden distance formula. The velocity of the object movement from frame to frame is calculated by using the distance and frame rate of the recorded video. The velocity is stored in the array. Velocity of moving object is in 2-dimension (since camera is static). Evaluation is now performed on TMS 320C6414T DSK and the values are visualised in the figured. Figure 5.1 shows the step by step process of tracking a moving object. From this figure it is clear that the inputs and the output produced at each step.

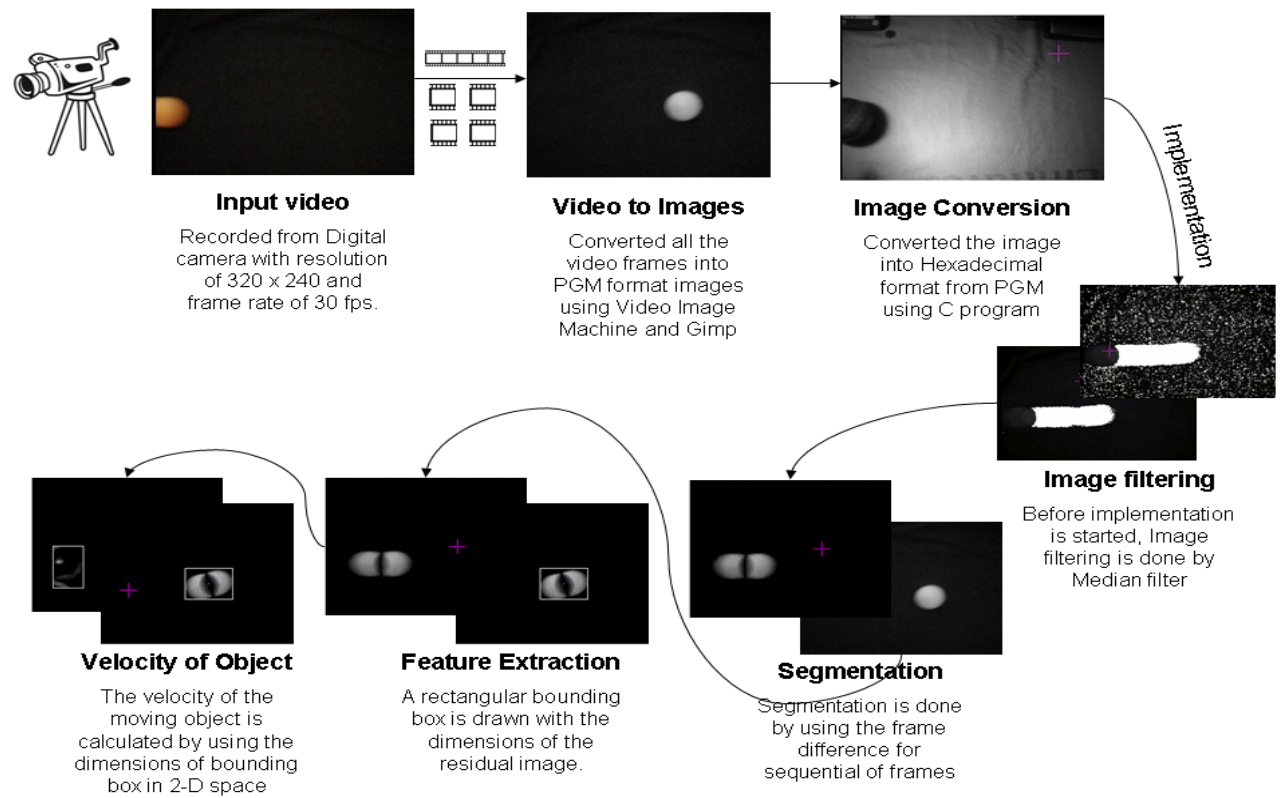


Figure 5.1: Step by step diagrams during evaluation and implementation



## Chapter 6

# Conclusion

Tracking of moving object is evaluated on TMS320C6416T DSK which is a major application in security, surveillance and vision analysis. The TMS320C6416T DSK external memory is used to store the imported images on board and the implementation is done. The implementation is performed by removing noise in image and separating foreground and background objects. The object is visualized and its centroid is calculated. The distance it moved between frame to frame is stored and using this velocity is calculated with the frame rate of video. The velocity of moving object in 2-dimension (since the camera is static) is determined.

In future, algorithms can be implemented on other hardware devices. These algorithms can also be extended for the used of real-time applications and object classifications.

# Bibliography

- [1] Gimp software. <http://www.gimp.org/downloads/>.
- [2] Image video machine software. [http://www.download.com/Image-Video-Machine/3000-2186\\_4-10359035.html](http://www.download.com/Image-Video-Machine/3000-2186_4-10359035.html).
- [3] Ulead video studio software. <http://www.ulead.com/vs/>.
- [4] Tms320c6416t dsk technical reference. Technical Report 508035-0001 Rev. A, November 2004.
- [5] C. Kamath A. Gyaourova and S.-C. Cheung. Block matching for object tracking. <https://computation.llnl.gov/casc/sapphire/pubs/UCRL-TR-200271.pdf>, October 2003.
- [6] David S. Bright. Removing shot noise from face image. <http://www.nist.gov/lispix/imlab/noise/shotfc.html>, May 25 2004.
- [7] Berthold K. P. Horn and Brian G. Schunck. Determining optical flow. *ARTIFICIAL INTELLIGENCE*, pages 389–407, 1992.
- [8] Jan F. Jørgensen. Mean filter in world wide web. [http://www.imagemet.com/WebHelp/hid\\_filters\\_smoothing\\_mean.htm](http://www.imagemet.com/WebHelp/hid_filters_smoothing_mean.htm).
- [9] Martin Mangard. Motion detection on embedded smart cameras. Master’s thesis, Technische Universität Graz, 2006.
- [10] Ashley Walker Robert Fisher, Simon Perkins and Erik Wolfart. Gaussian smoothing in world wide web. <http://homepages.inf.ed.ac.uk/rbf/HIPR2/gsmooth.htm>, 2003.
- [11] Ashley Walker Robert Fisher, Simon Perkins and Erik Wolfart. Mean filter in world wide web. <http://homepages.inf.ed.ac.uk/rbf/HIPR2/mean.htm>, 2003.
- [12] Dr. Mike Spann. Image segmentation in world wide web. <http://www.eee.bham.ac.uk/spannm/Teaching%20docs/Computer%20Vision%20Course/Image%20Segmentation.ppt>, Course resources for 2008.
- [13] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38(4):13, 2006.
- [14] Y. T. Zhou, V. Venkateswar, and R. Chellappa. Edge detection and linear feature extraction using a 2-d random field model. *IEEE Trans. Pattern Anal. Mach. Intell.*, 11(1):84–95, 1989.