Development, Implementation and Optimization of Real Time Target Tracking Algorithms on BeagleBoard-xM

Srikanth A[1]

[1]Research Associate, Department of Research and Development
Nitte Meenakshi Institute of Technology, Yalahanka, Bangalore, Karnataka, India
sria91@gmail.com

Abstract. This work discusses the development and realization of intelligent computer vision based target tracking algorithms on the BeagleBoard-xM ARM based embedded platform, which tracks moving targets in a continuous scene operating in real-time. The integration level of embedded platform is given a very high significance throughout the design. Combining the embedded platform with target tracking, have many end uses especially in robotics, surveillance, human-computer interaction applications, etc. The results of the implementation on BeagleBoard-xM are compared with the implementation on an Intel Core i3 platform.

Keywords: Embedded Computer Vision, Correlation based Target Tracking, ARM, BeagleBoard-xM, Linux

1 Introduction

In recent years the technique of moving target detection and tracking is increasingly becoming more and more popular and has rapidly developed into a very important prospect in the field of computer vision.

The basic idea of tracking target is to register image from definition. Image registration [6] is the process of overlaying images (two or more) of the same scene taken at different times, from different viewpoints, and/or by different sensors. Registration aligns two images – the template and search frame, the template is searched in the search frame using best suitable algorithm. In the literature [7] makes use of BeagleBoard-xM as the embedded platform. Background subtraction technique is made use of to detect the moving object. A color histogram is computed using the HSV model and Bhattacharya distance is used for similarity measurement. They have not utilized all the available resources of the BeagleBoard-xM including the DSP. The algorithm runs at 400ms per frame. [8] uses an ARM9 based S3C2440 chip as the core processor. They use a segmentation approach for moving target detection. The tracking is accomplished by using the Mean-Shift algorithm. They could achieve an execution time of 100ms.

Our work contributes to this line by providing a brief review of some of the image registration algorithms applied to target tracking, implementing and analyzing their performance using a suitable image processing hardware. The performance of the algorithms implemented on the BeagleBoard-xM is also compared with the implementation on an Intel Core i3 platform.

2 System Overview

Vision often uses complex, computationally demanding algorithms; implementing these under severe cost, size, and energy constraints requires selecting the right processor for the job and requires to optimize algorithm implementations for the selected processor. The previous object tracking systems are mostly built using general purpose desktop PC or Laptop, which is inadequate to meet the demands of real-time computation and miniaturization. In contrast, an embedded system has limited processing power and limited memory resources. The performance can be improved by selecting the proper mobile processor in addition to an environment having different on-chip resources.

Low-power ARM based embedded system-on-chips (SoCs) combine various co-processors including a Vectorized Floating Point Unit (FPU), a Graphics Processing Unit (GPU) and a Digital Signal Processor (DSP) on a single chip. In the literature [7-8], they have been successfully used to realize tracking algorithms on hardware. Hence, they can be an ideal solution for implementing computer vision based algorithms such as Target Tracking. The embedded platform which has been selected for the study is BeagleBoard-xM [1]. The results are also compared with that of an Intel PC platform.

2.1 Embedded Platform

The BeagleBoard-xM [1] is a low-cost, low power, fan-less open source hardware single board computer produced by Texas Instrument in association with Digi-Key. BeagleBoard-xM is the modified version of BeagleBoard which has faster CPU core (clocked at 1GHz compared to 720MHz) and more RAM (512 MiB compare to 256 MiB). The BeagleBoard-xM was designed with open source development in mind and as a way of demonstrating the Texas Instruments DM3730 system-on-a-chip.

The board uses up to 2 W of power and because of the low power consumption, no additional cooling and heat sinks are required. By eliminating all of the on-board peripherals and by providing standard expansion buses like high-speed USB 2.0, Ethernet port and HDMI port, developers and researchers can bring their own peripherals and expand the board ability what they want. It has been equipped with a minimum set of features to allow the user to experience the power of the processor.

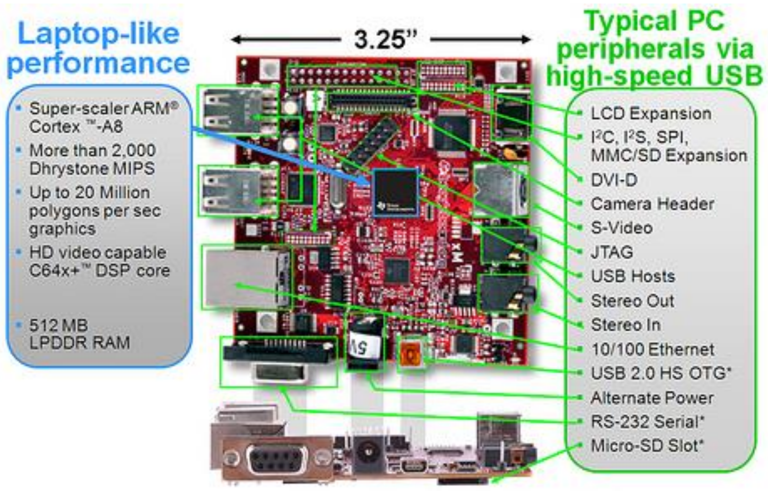

Laptop-like performance
Super-scaler ARM® Cortex ™-A8
More than 2,000 Dhrystone MIPS
Up to 20 Million polygons per sec graphics
HD video capable C64x+™ DSP core
512 MB LPDDR RAM
3.25"
Typical PC peripherals via high-speed USB
LCD Expansion
I²C, I²S, SPI, MMC/SD Expansion
DVI-D
Camera Header
S-Video
JTAG
USB Hosts
Stereo Out
Stereo In
10/100 Ethernet
USB 2.0 HS OTG*
Alternate Power
RS-232 Serial*
Micro-SD Slot*

Fig. 1. BeagleBoard-xM

2.2 Intel Platform

The Target Tracking algorithms are implemented on an Intel Core i3 M 350 Laptop running Windows 7. The system is clocked at 2.27 GHz, having 3 GB of RAM. Visual Studio is used as the development environment. The algorithms are written in standard C++ and the compared metrics are recorded.

2.3 Simple DirectMedia Layer (SDL)

Simple DirectMedia Layer is a cross-platform free and open source multimedia library designed to provide simpler low level access to audio, keyboard, mouse, joystick, graphics devices. SDL has the word "layer" in its title because it is actually a wrapper around operating-system-specific functions. The main purpose of SDL is to provide a common framework for accessing these functions. SDL has been used before to write computer games and other multimedia applications that run on many operating systems.

2.4 GStreamer

GStreamer is a pipeline-based multimedia framework written in C with type system based on GObject. GStreamer allows a programmer to create a variety of media-handling components, including simple audio playback, audio and video playback, recording, streaming and editing. The pipeline design serves as a base to create many types of multimedia applications such as video editors, streaming media broadcasters and media players.

2.5 Digital Video Software Development Kit (DVSDK)

DVSDK is a set of software and data for development targeting for misc TI processors. DVSDK enable DaVinci system integrators to quickly develop Linux-based multimedia applications.

3 Target Tracking

The general approach for tracking using data from imaging sensors (camera) is as follows. The target to be tracked is first specified by a human operator. An image registration algorithm then searches for the target in each subsequent image obtained by the imaging sensor. The measurement resulting from the image registration algorithm is used to find the position of the target. The key issues in image registration are the time required for registration and the accuracy of registration, i.e., the measure of how close is the match between sub image in the search space and the reference image.

Image Registration is a process, which finds the location where optimal matching is obtained by matching a template image called the reference image over the searching region of an input image using a suitable similarity measures. The method although computationally intensive, is simple, straightforward and robust and requires no a priori information about the two images.

3.1 Search Strategies

Fine Search. In fine search strategy, registration starts at the top left corner of the search space and continues along each row and column moving the sub image by one pixel each time. The whole search image will be evaluated in this method. The accuracy of registration algorithm using fine search is good but the computation time is large.

Coarse – to – Fine Search. In coarse-fine search strategy, registration is done by extracting sub images of equal size as that of the reference image at the start of search space on a coarse grid at every n¬th point. An approximate match point is found at the end of this step. A full search is done in a local region surrounding this match point. A coarse-fine search strategy is efficient if the system allows a minor degradation in accuracy.

Around the Object Fine Search. In this search strategy, the fine search is performed around the area of selected template.

Among these the fine search is more efficient but it is time-consuming. Coarse-fine search is moderately efficient and consumes lesser time for each frame. Fine search around the object is efficient only when the object stays within a certain boundary-limited window around the matched position of the previous frame but the time consumed is greatly reduced and hence has been explored in this work.

Following algorithms have been studied which are based on similarity measure (correlation) between the reference image and search space. Correlation Coefficient is the classical representative of the area-based methods. It matches directly image intensities, without any structural analysis. They are sensitive to the intensity changes, introduced by noise, varying illumination, and/or by using different sensor types.

1. Pearson Correlation Coefficient [9].
2. Spearman Correlation Coefficient [10].
3. Cross Correlation (CC) [11].
4. Normalized Cross Correlation (NCC) [12] and [13].
5. Normalized Area Correlation Metric (NACM).
6. Cross Sectional Histogram Correlation (CHC) [14].

4 Overview of the Tracking System

The embedded target tracking system (Fig 2) based on BeagleBoard-xM consists of four different modules: Image Acquisition module, Pre-processing module, Tracking module and the Display and User Interface module.

4.1 Image Acquisition Module

The images are acquired from a live camera feed or a stored video file (used for testing) at 320 x 240 pixels and 30 fps. The camera gets connected to the BeagleBoard-xM via one of its USB ports. GStreamer [2] multimedia framework has been used in order to access the frames from the image source. The GStreamer framework provides a unified way of accessing the V4L2 camera and the stored video by setting up a proper pipeline. This forms the image acquisition module.

4.2 Pre-processing Module

In order to reduce the computation required for processing each frame, the input image sequence is converted into grayscale by the preprocessing module.

4.3 Display and User Interface Module

Input frames acquired are continuously displayed on a display system attached to the board. The tracking is initiated by the user through a mouse click on any part of the input image being displayed. A template is extracted around the clicked co-ordinate and is used as reference for the tracking module. The tracked target is also annotated on the displayed image by a colored rectangle. Simple DirectMedia Layer (SDL) [3] is used to implement this module due to it being simple and light on resources which is essential in any embedded system. The SDL framework also provides the capability for handling keyboard and mouse events.

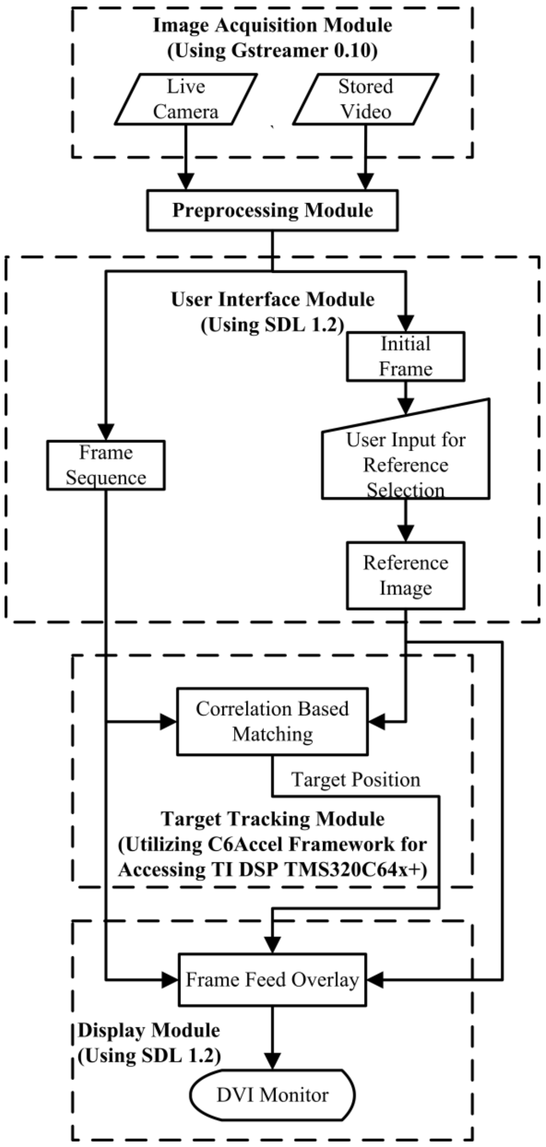


Fig. 2. Block diagram of the Target Tracking System.

4.4 Tracking Module

The tracking module performs the actual work of continuously matching the reference template with the portions of the input frames using the fine search around-the-object approach discussed in the last section. The tracking module exploits the DSP core (TI's TMS320C64x+) onboard the BeagleBoard-xM to offload the computationally intensive task of evaluating the correlation between the reference template and the search sub-images of the input frame in order to obtain the best match for the 2D target position. C6Accel [4] framework has been used to develop the DSP kernels for all the tracking algorithms discussed. Fig 3 shows the overview of the C6Accel framework.

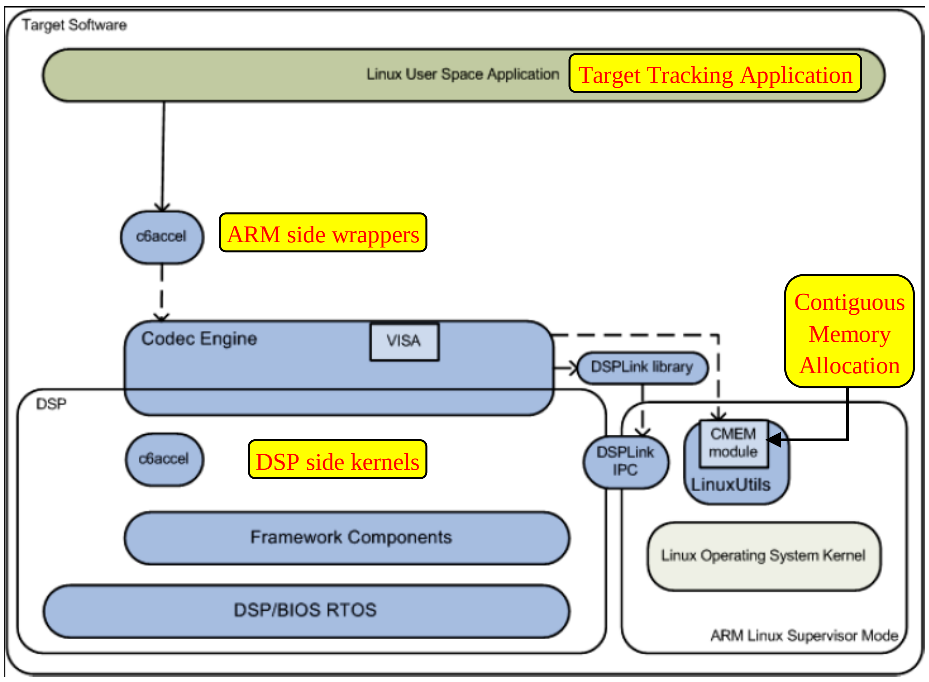


Fig. 3. The interface between the ARM and the DSP

5 System Setup and Implementation

A minimal version of Angstrom Linux 2011.03 is setup on the BeagleBoard-xM. Angstrom Linux is chosen as it has good support for DSP development using the C6Accel framework. It also allows for high performance by utilizing the complete 1GHz clock frequency of the board and provides with a stable kernel. The customized software image is generated using Narcissus [5] and installed on the SD card from which the board boots up. The necessary toolchains (gcc), software libraries (sdl, gstreamer) along with their development headers and kernel modules (dsplink, cmem, uvcvideo) are selected to be integrated into the generated image to save the burden of manual compilation and installation of the same later.

The Digital Video Software Development Kit (DVSDK) from TI is installed on a 32-bit intel PC with Ubuntu 12.10 for accessing the C6Accel framework. C6Accel framework is used for writing the algorithms as DSP kernel, which gets compiled using TI CodeGen tools into a DSP side executable. The ARM side wrapper functions are also written and their static libraries are cross compiled on the PC. The generated libraries are finally linked with the target tracking application on the board. The output of the system can be displayed on a monitor via the DVI-D output. A USB keyboard and a mouse are connected to the system for user input as shown in Fig 4.

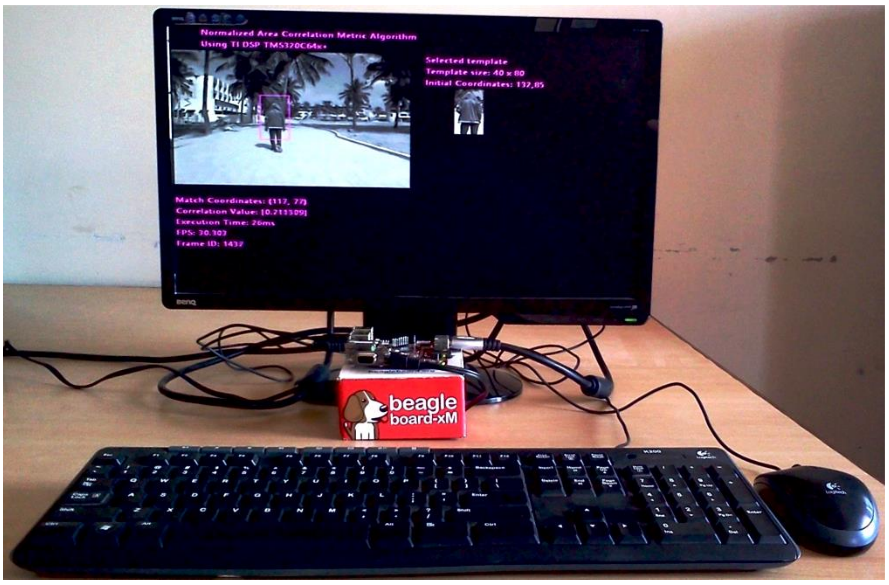

Normalized Area Correlation Metric Algorithm
Using TI DSP TMS320C64x+
Selected template
Template size: 40 x 80
Initial Coordinates: 132,85
Match Coordinates: (117, 77)
Correlation Value: [0.211509]
Execution Time: 26ms
FPS: 30.303
Frame ID: 1437
BenQ
beagle board-xM

Fig. 4. The laboratory setup of Target Tracking System.

The Linux kernel is configured to set aside a part of RAM for shared use by ARM and the DSP. The CMEM module initializes this block of RAM since the data interchanged with the DSP needs to be physically contiguous. On execution the target tracking application loads the DSP side executable. The target tracking application allocates the memory in the shared region for the input, reference images as well as other input and output arguments for the kernel. The application the gives a call to the DSP side algorithm kernel using the ARM side wrapper function. The whole process is demonstrated in Fig 3.

6 Pseudocode for CHC Algorithm

```
CHC(S, m1, n1, T, m, n, xp, yp)
// Inputs
// S = input image
// m1, n1 = input image width and height
// T = reference image
// m, n = reference image width and height
// xp, yp = previous target co-ordinates
// Outputs
// xm, ym = target co-ordinates after matching
// Constants
// Ww = width of search window
// Wh = height of search window

// Algorithm
p = compute row histogram of T (eqn 1)
y = compute col histogram of T (eqn 2)
Tm = compute mean of T (eqn 3)

match_corr = 'zero'

for x = prev_x - Ww/2 to prev_x + Ww/2
  for y = prev_y - Wh/2 to prev_y + Wh/2
    R = compute row histogram of sub block of S (eqn 4)
    C = compute col histogram of sub block of S (eqn 5)
    Tm = compute mean of sub block of S (eqn 6)
    Lp = compute row correlation between S and T (eqn 7)
    Ly = compute col correlation between S and T (eqn 8)
    corr = compute CHC between S and T (eqn 9)
    if corr > match_corr
      match_corr = corr
      xm = x
      ym = y
    endif
  endfor
endfor
```

Reference row histogram
$$p_i = \frac{1}{n}\sum_{k=0}^{n-1} T_{(i,k)} \tag{1}$$

Reference column histogram
$$\gamma_j = \frac{1}{m}\sum_{l=0}^{m-1} T_{(i,j)} \tag{2}$$

Reference mean
$$\overline{T} = \frac{1}{nxm}\sum_{(i=0,j=0)}^{(n-1,m-1)} T_{(i,j)} \tag{3}$$

Input row histogram
$$R_i = \frac{1}{n}\sum_{k=0}^{n-1} S_{(y+i,x+k)} \tag{4}$$

Input column histogram
$$C_j = \frac{1}{m}\sum_{l=0}^{m-1} S_{(x+i,y+j)} \tag{5}$$

Input mean $$\overline{S} = \frac{1}{nxm}\sum_{(i=0,j=0)}^{(n-1,m-1)} S_{(x+i,y+j)} \quad (6)$$

Row correlation $$\lambda_{\rho} = \frac{\sum_{i=0}^{m-1}\left[\left(p_i - \overline{T}\right)\left(R_i - \overline{S}\right)\right]}{\sqrt{\left(\sum_{i=0}^{m-1}\left(p_i - \overline{T}\right)^2 \sum_{i=0}^{m-1}\left(R_i - \overline{S}\right)^2\right)}} \quad (7)$$

Column correlation $$\lambda_{\gamma} = \frac{\sum_{j=0}^{n-1}\left[\left(\gamma_j - \overline{T}\right)\left(C_j - \overline{S}\right)\right]}{\sqrt{\left(\sum_{j=0}^{n-1}\left(\gamma_j - \overline{T}\right)^2 \sum_{j=0}^{n-1}\left(C_j - \overline{S}\right)^2\right)}} \quad (8)$$

Cross Sectional Histogram Correlation $$\sigma(x,y) = \lambda_{\rho}^2 \times \lambda_{\gamma}^2 \quad (9)$$

7 Results and Discussion

The execution times for the algorithms on BeagleBoard-xM (without and with the DSP) and on Intel Core i3 platform are compared in Table 1. The snapshots of the result along with the metrics used for comparison are available in Table 2–5.

The performance of the algorithms on BeagleBoard-xM platform is improved up to 5 times through optimization when compared to an Intel platform though the algorithms still don't operate in real-time. The performance is further improved 2.3 to 4.6 times by utilizing the computing power of the onboard DSP core and nearly all the algorithms have met the requirements of real-time. The comparatively lower performance of the CHC is justified by the necessity for computation of the histogram for each sub-image.

Table 1. Comparison of execution time for different algorithms.

Algorithm Name	**Execution time in millisecond**			
	Intel Core i3	**BeagleBoard-xM**		
		ARM (Unoptimized)	**ARM (Optimized)**	**DSP (Optimized)**
PEARSON	80	256	56	21
SPEARMAN	210	154	42	9
CC	115	204	50	19
NCC	160	334	45	19
NACM	190	754	81	25
CHC	110	1580	130	94

Acknowledgements. The author expresses their sincere gratitude to Prof N R Shetty, Director, Nitte Meenakshi Institute of Technology; Dr H C Nagaraj, Principal, Nitte Meenakshi Institute of Technology and Dr Jharna Majumdar, Former Scientist 'G' DRDO, Dean R&D, Prof & Head, Department of CSE (PG), Nitte Meenakshi Institute of Technology for providing the support, encouragement and infrastructure to carry out the research. The author would like to thank Mr Venkatesh G M, Research Associate, Nitte Meenakshi Institute of Technology for generating the results on the Intel platform and providing for the comparison. The author would also like to acknowledge the Robotics Team of Nitte Meenakshi Institute of Technology for their support.

References

1. BeagleBoard-xM Rev C System Reference Manual, http://www.beagleboard.org
2. GStreamer Manual, http://www.gstreamer.net
3. Simple DirectMedia Layer (SDL) Tutorials, http://www.libsdl.org
4. C6Accel Advanced Users Guide, http://processors.wiki.ti.com/index.php?title=C6Accel_Advanced_Users_Guide
5. Narcissus, http://narcissus.angstrom-distribution.org
6. Zitova B., Flusser J.: Image registration methods: a survey. In: Image and Vision Computing 21, pp. 977-1000. Elsevier (2003)
7. Khang S., Yap V.V., Sebastian P.: Implementation and Optimization of Human Tracking System using ARM Embedded Platform. In: 4th International Conference on Intelligent and Advanced Systems, Vol 1, pp. 353-356. (2012)
8. Pan L., Liu X.: The Study of Target Tracking Based on ARM Embedded Platform. In: Journal of Computers, Vol 7, No 8, pp. 2015-2023, (2012)
9. Soares F., Affonso L., Diego.: Application of Fourier Descriptors and Pearson Correlation for Fault Detection in Sucker Rod Pumping System. In: IEEE Conference on Emerging Technologies and Factory Automation. (2009)
10. Liu D., Yeung S., Sun D.M., Qiu Z.D.: A Spearman Correlation Coefficient Ranking for Matching-score Fusion on Speaker Recognition. In: IEEE Region 10 Conference (2010)
11. Didon J.P., Langevin F.: Registration of MR Images: From 2D to 3D, using a Projection Based Cross Correlation Method. In: 17th IEEE Annual Conference on Engineering in Medicine and Biology Society. (1995)
12. Bunting P., Lucas R., Labrose F.: An Area Based technique for Image-to-Image registration of Multi-Modal remote sensing data. In: IEEE International Geoscience and Remote Sensing Symposium. (2008)
13. Patnaik S., Bombaywala S., Sarvaiah J.N.: Image Registration by template matching using Normalized Cross-Correlation. In: International Conference on Advances in Computing, Control and Telecommunication Technologies. (2009)
14. Takei R.: A New Grey-Scale Template Image Matching Algorithm using the Cross-Sectional Histogram Correlation Method. (2003)

Table 2. Results of the different tracking algorithms on test video 1

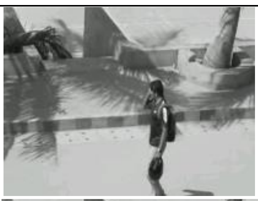
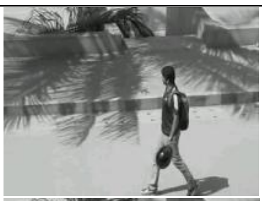
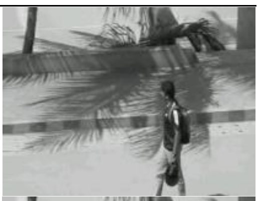
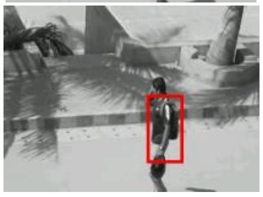
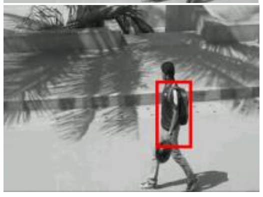
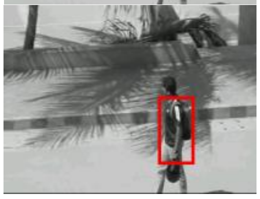
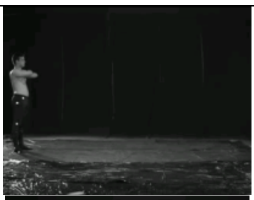
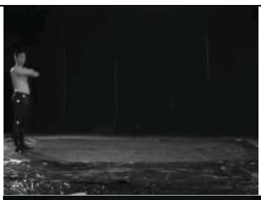
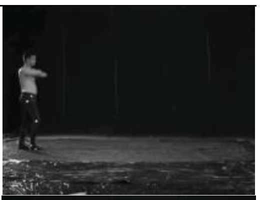
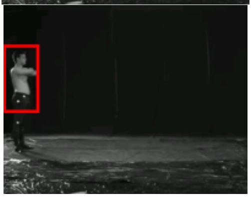
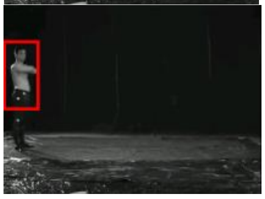
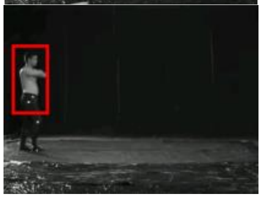
Frame ID	**1**	**11**	**21**
Input Frames			
Output Frames			
PEARSON	(201,148)[1] [.6397][2]	(213,132) [.7081]	(218,154) [.5045]
SPEARMAN	(201,150) [.9990]	(213,142) [.9998]	(217,161) [.9900]
CC	(201,150) [.9490]	(213,142) [.9542]	(217,161) [.9347]
NCC	(201,148) [.8251]	(213,132) [.7697]	(217,151) [.7181]
NACM	(201,150) [.7932]	(213,142) [.8156]	(217,161) [.7312]
CHC	(201,148) [.9781]	(213,133) [.9605]	(218,151) [.9682]

Table 3. Results of the different tracking algorithms on test video 2

Frame ID	**1**	**11**	**21**
Input Frames			
Output Frames			
PEARSON	(22,94) [.7305]	(22,94) [.7022]	(33,95) [.6079]
SPEARMAN	(22,94) [.9999]	(22,94) [.9980]	(33,95) [.9997]
CC	(22,94) [.9992]	(22,94) [.9771]	(33,95) [.9506]
NCC	(22,94) [.9663]	(22,94) [.9383]	(33,95) [.9216]
NACM	(22,94) [.9983]	(22,94) [.9520]	(33,95) [.8971]
CHC	(22,94) [.9976]	(22,94) [.9961]	(33,95) [.9923]

1 Target location in pixel co-ordinates (x, y)
2 Correlation value of the match

Table 4. Results of the different tracking algorithms on test video 3

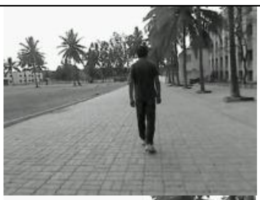
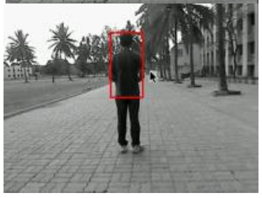
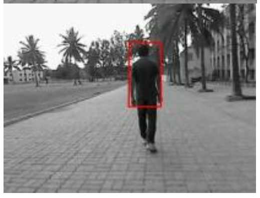
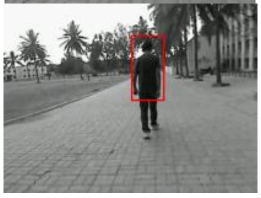
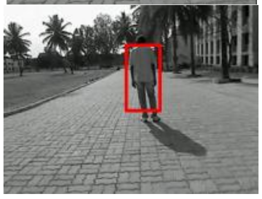
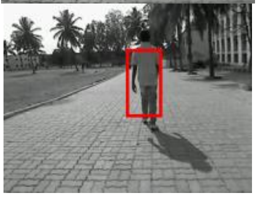
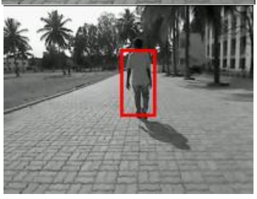
Frame ID	1	11	21
Input Frames			
Output Frames			
PEARSON	(154,91) [.4047]	(176,100) [.3232]	(181,96) [.3670]
SPEARMAN	(154,91) [.9994]	(177,100) [.9986]	(181,96) [.9981]
CC	(154,91) [.9072]	(177,100) [.8251]	(181,96) [.8065]
NCC	(154,91) [.7937]	(177,100) [.5984]	(181,96) [.5577]
NACM	(154,91) [.7937]	(177,100) [.5984]	(181,96) [.5577]
CHC	(154,89) [.9430]	(178,100) [.9315]	(181,95) [.9145]

Table 5. Results of the different tracking algorithms on test video 4

Frame ID	1	11	21
Input Frames			
Output Frames			
PEARSON	(173,94) [.7770]	(178,100) [.5525]	(172,98) [.5473]
SPEARMAN	(174,93) [.9997]	(178,100) [.9989]	(172,98) [.9986]
CC	(174,93) [.9800]	(178,100) [.9275]	(172,98) [.9195]
NCC	(173,94) [.9032]	(178,100) [.6221]	(172,98) [.5858]
NACM	(174,93) [.9032]	(178,100) [.6221]	(172,98) [.5858]
CHC	(174,93) [.9960]	(178,96) [.9777]	(171,94) [.9819]