Implementation of Real Time Local Search Particle Filter Based Tracking Algorithms on BeagleBoard-xM

Jharna Majumdar[1], Amar Mani Aryal[2], Nabin Sharma Rijal[2], Parashar Dhakal[2], Nilesh Kumar Mishra[2]

[1]Dean R&D, Prof. & Head, Department of CSE (PG)
NitteMeenakshi Institute of Technology, Bangalore-560064, India

[2]Final Year, Undergraduate students, Department of ECE
NitteMeenakshi Institute of Technology, Bangalore-560064, India

Abstract

This work hybridizes Particle Filter and Local Search algorithm for the realization of intelligent computer vision based target tracking on BeagleBoard-xM, an ARM based embedded platform, which tracks moving targets in a continuous scene operating in real-time. The integration of Local Search algorithms with Particle filter significantly increases the accuracy of tracking. The use of embedded board helps to reduce the space requirement and the cost of design. The implementation of target tracking on embedded platform has many end uses especially in the field of robotics, surveillance, human-computer interaction applications, etc.

Keywords: *ARM, BeagleBoard–xM, Color histogram, Distance Measures, Embedded Computer Vision, GStreamer, Linux, Local Search, Particle filter, Re-Sampling, SDL, Target Tracking.*

1. Introduction

Object tracking, an inevitable constituent of computer vision, generates the trajectory of the specified object over time by locating its position in each frame in the video sequence. The use of object tracking is pertinent in various vision applications such as motion based recognition, automated surveillance, video indexing, human computer interface, vehicle navigation, road traffic control, and security and surveillance systems.

Particle filter has as recently as 1993 surfaced the domain of computer vision and has become very useful in computer vision related applications. The advantage which the particle filter has over other types of filters like Kalman filter, Extended Kalman filter, etc. is that it allows a state space representation of any distribution. It also handles nonlinear, non-Gaussian, dynamical and observation models, and nonlinear, non-Gaussian process and observation noises [7].The introduction of local search [6] based scheme increases the accuracy of tracking since predictions are refined in a local search procedure that utilizes the most recent observation.

The present work uses BeagleBoard-xM. To achieve real time performance, DSP core i.e. TI's TMS320C64x+ in the board is used to offload the computationally intensive task of evaluating distance measure. The ARM CORTEX-A8 CPU in the board is used to realize frame acquisition and display modules. Tracking is initiated by the user by giving a click on the part of the input image containing the object to be tracked. A template is extracted around the clicked co-ordinate and is stored as reference image. A color histogram is computed using the HSV model and the weight of the particle is calculated comparing the particle histogram with the reference histogram using one of the distance measures. The position of the particle having highest weight is chosen as the best match from the Particle filter algorithm. The best match from the Particle filter is applied to the local search procedure which later yields a refined solution.

Our work contributes to this line of real time tracking by providing a detailed review of the Particle filter algorithm applied to target tracking and the improvement in the accuracy of the tracking after introducing Local Search scheme. The performance of the work is analyzed with and without utilizing DSP supplied with the board. A comparative analysis is also done implementing basic Particle filter alone and Particle filter with Local Search scheme. The Local Search Particle filter [6] comprises of distance measures, local search procedure and re-sampling methods, which are varied on an Intel platform and compared using accuracy method. The best methods and the algorithm giving the best performance are implemented on embedded platform.

2. Platform Overview and Specifications

Computer vision often uses complex, computationally intensive algorithms with constraints like high cost, size,

and energy considerations and hence selection of the right processor to optimize algorithm implementation is important. The object tracking systems of the past are mostly built using general purpose desktop PC or laptop, which is inadequate to meet the demands of real-time computation and miniaturization. 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. The embedded platform which has been selected for the study is BeagleBoard-xM [1].

2.1 Embedded Platform

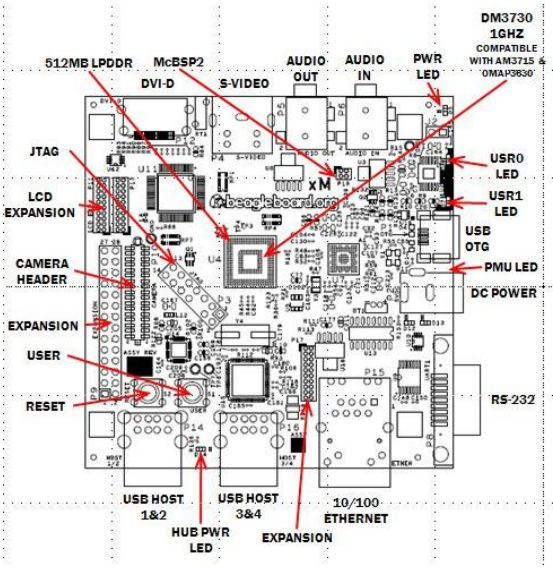


Fig. 1 Schematic of BeagleBoard-xM

The BeagleBoard-xM [1] is a low-cost, low power, fan-less open source hardware single board computer with DM3730 system-on-a-chip. 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 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 the on-board peripherals and 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 whatever they want. A minimal version of Linux Angstrom is installed in the board to experience the power of processor.

3. Local Search Particle Filter Based Tracking

The general approach for the proposed tracking involves the acquisition of data using imaging sensors (camera), initialization of the target to be tracked by the user, the use of Particle filter algorithm to search the target in each subsequent frames and Local Search refinement [6] to improve the accuracy of tracking. The data resulting from the algorithm is used to find the position of the target. The Local Search Particle Filter based tracking [6] involves the use of distance measures (to measure histogram similarity), scheme for Local Search and re-sampling methods (to solve particle degeneracy).

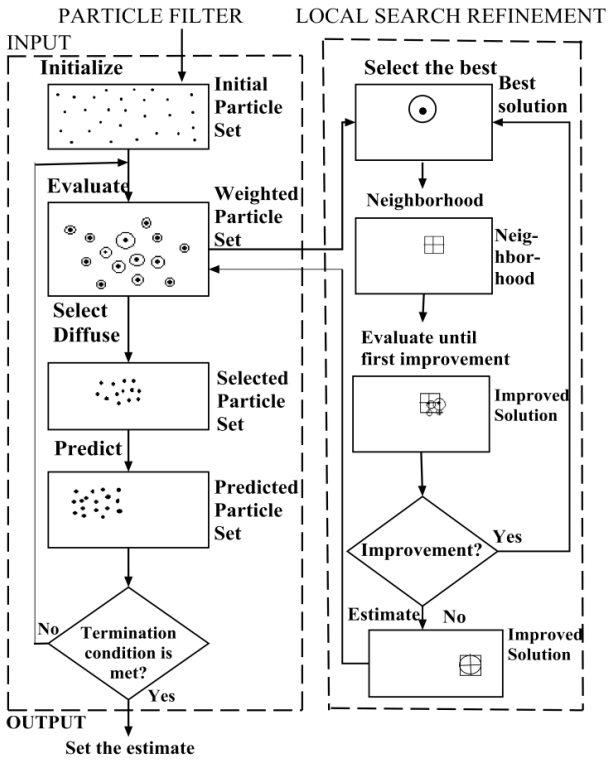


Fig. 2 Flowchart for Local Search Particle Filter Scheme.

3.1 Particle Filter

The key idea of particle filtering is to represent the posterior probability density function by a set of discrete samples known as particles. Each particle represents a

hypothesis of the state and it is randomly drawn from the prior density. In other words, each particle is a random state representing one possible location of the object being tracked. The set of particles contains more weight at locations where the object being tracked is more likely to be present. We can determine the trajectory of the tracked object by taking the particle with the highest weight or the weighted mean of the particle set at each time step.

The weight of each particle is computed in the consecutive frames using distance measure between the particle and reference histogram. Re-sampling methods are used to select and re-generate the particle to keep tracking and particle count intact. This is done to reduce the degeneracy problem in particle filters. The weight characterizes the quality of a specific particle. A large weight will be assigned to a good particle, and a small weight will be assigned to a bad particle.

3.2 Local Search Scheme

In the Local Search [6] scheme, the highest weight particle obtained in the Particle filter stage is refined using a local search procedure as shown in Fig. 2. The neighborhood around the highest weight particle is defined as shown in Fig. 3(a) and the weight is computed taking the neighborhood location and comparing the corresponding particle histogram with the reference histogram. The weight calculated is then compared with highest weight obtained from the Particle filter stage. If the weight of the neighboring particle is greater than that of the weight computed from Particle filter stage, it is taken as the improved solution. The procedure is repeated for all the neighbors considered in a predefined order as shown in Fig. 3(a) until the first improvement in the weight is observed as shown in Fig. 3(b). The improved particle is now taken as the reference particle as shown in Fig. 4(b) and the above process is repeated. The repetition is continued until there is no improvement in the weight as shown in Fig. 4(d) and the target position is updated corresponding to the new particle as shown in Fig. 4(d).

Fig.3 shows the Local Search strategy [6]. Our work makes use of 8 neighboring pixels around the highest match co-ordinate from the Particle filter for the refinement purpose.

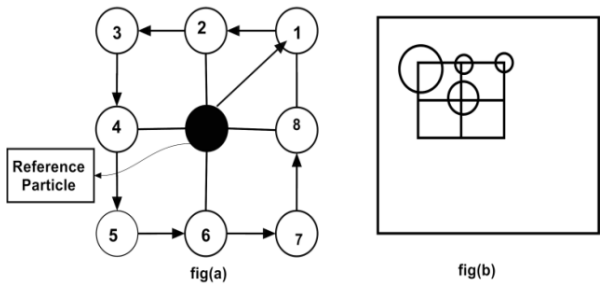


Fig. 3 Local Search Strategy

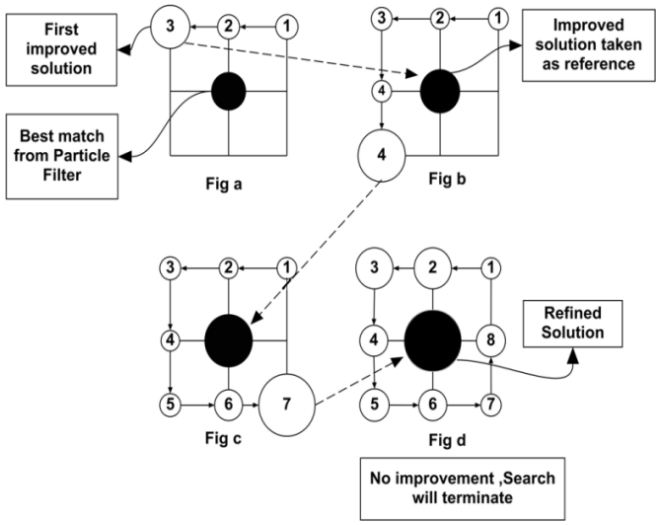


Fig. 4 Detailed overview of Local Search scheme
[Larger circle indicate larger value of weight]

3.3 Re-sampling

Re-sampling [5] is used to solve the problem of particle degeneracy by which particles are re-generated around the target location to make the tracking process intact. Residual re-sampling was selected owing to its performance.

Residual re-sampling, also called remainder re-sampling, is an efficient way of decreasing the variance due to re-sampling. Residual re-sampling involves calculating the sum of the weight of the particles, normalizing the weights with the calculated sum, selecting the particle with normalized weights greater than the predefined threshold and truncating those having weights less than the threshold and replicating the particles (equal to the number of truncated particles) around the particle with maximum weight.

3.4 Distance Measures

A number of distance measures are used to compute the weight of the particle. The value of distance ('d') lies between 0 and 1. For a perfect match, value of'd' should be close to 1.

3.4.1 Bhattacharya Coefficient

$$\rho[H1,H2] = \sum_{u=1}^{m} \sqrt{H_{1u} * H_{2u}} \quad (1)$$

Where, u =histogram bin index, m=number of bins.

For two identical normalized histograms we obtain $\rho = 1$, indicating a perfect match. To quantify the distance between two distributions, the distance'd' is defined as

$$d = \sqrt{1 - \rho[H1,H2]} \quad (2)$$

3.4.2 Correlation Coefficient Histogram Method

$$d(H_1,H_2) = \frac{\sum H'_1(i) \cdot H'_2(i)}{\sqrt{\sum H'^{2}_1(i) \cdot H'^{2}_2(i)}} \quad (3)$$

3.4.3 Chi-Square Distance

$$d(H_1,H_2) = \sum \frac{(H_1(i) - H_2(i))^2}{H_1(i) + H_2(i)} \quad (4)$$

3.4.4 Intersection Distance

$$d(H_1,H_2) = \sum \min(H_1(i), H_2(i)) \quad (5)$$

Where, H_1 = Reference Histogram
H_2 = Particle Histogram

$$H'_1 = H_1 - \overline{H_1}$$

$$H'_2 = H_2 - \overline{H_2}$$

To choose the best combination of distance measure and re-sampling methods accuracy measures are used. These methods consider consecutive frames, and repeatedly implement various algorithms to compute the accuracy of target tracking. The various accuracy methods used in the work are:-

1. Pixel Cross Correlation
2. Pixel Difference

4. The Proposed Algorithm

Step1: Initialize initial position (X_t) of target in the first frame.

Step2: Generate a particle set of N particles
$\{X^m_t\}_{m=1....N}$ around the object or entire frame.

Step3: Calculate the histogram for all the particles and compute the weight by comparing with the reference histogram.

Step4: Normalize the weights by dividing weight of each particle by total weight.

$$w_{ni} = \left[\frac{w_i}{\sum_{m=1}^{N} w_m} \right]; i \in [1:N] \quad (6)$$

Where,

w_{ni} =Normalized weight of i^{th} particle

w_i = Weight of i^{th} particle

$\sum_{m=1}^{N} w_m$ = Sum of weight of all particles in current frame

N = no of particles

Step5: Select the particle location with highest weight as the match from Particle filter stage.

Step6: Apply local search procedure around the best match from step 5 to get refined solution.

Step7: Re-sample the particles for next iteration.

5. Overview of the Tracking System

The embedded target tracking system based on ARM consists of four different modules: Image Acquisition module, Pre-processing module, Tracking module and Display & User Interface module.

5.1 Image Acquisition module

The images are acquired from a live camera feed or a stored video file (used for testing) of 320 x 240 pixels at 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.

5.2 Pre-processing Module

In order to reduce the computation required for processing each frame and to reduce the light intensity variation problem, the input frames are converted into HSV using an efficient integer method by the preprocessing module.

5.3 Tracking Module

The tracking module performs the actual work of continuously tracking the reference template in the input frames. The tracking module exploits the DSP core (TI's TMS320C64x+) on the board which is used to calculate computationally intensive task of calculating the distance between the reference and particle histogram. C6Accel [4] framework has been used to develop the DSP kernels for all the distance measures, which can be called from the ARM side. The RGB to HSV conversion is also executed on DSP side.

5.4 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 by clicking 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 Direct Media Layer (SDL) [3] is used to implement this module. The SDL framework also provides the capability for handling keyboard and mouse events.

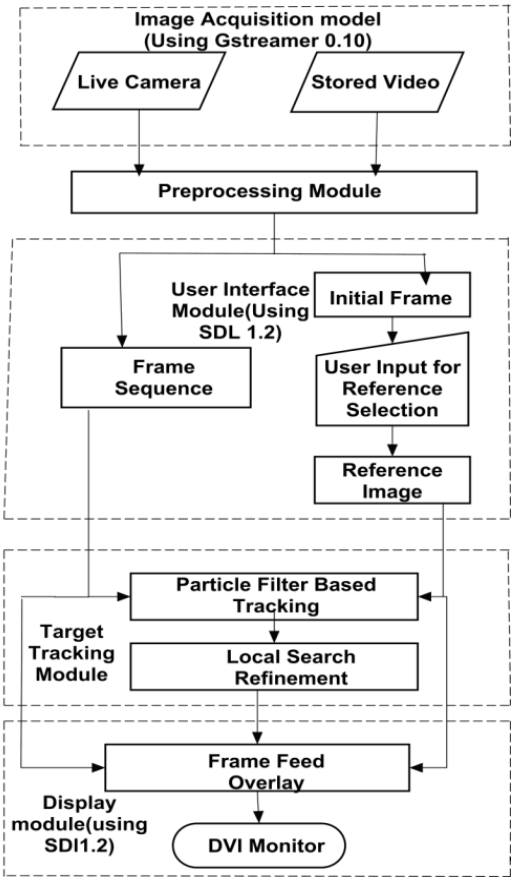


Fig. 5 Flow diagram of the Target Tracking system.

6. System Setup and Implementation

A minimal version of Angstrom Linux 2011.03 is installed on the BeagleBoard-xM. Angstrom Linux is chosen as it has good support for DSP development using the C6Accel framework. It also provides high performance by utilizing the complete 1GHz clock frequency of the board and a stable kernel. The customized software image is generated using Narcissus and is installed on the SD card from which the board boots up. The necessary tool chains (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 later manual compilation and installation of the same.

The Digital Video SDK from TI is installed on a 32-bit Intel PC with Ubuntu 12.10 for accessing the C6Accel framework used for compiling the DSP side kernels and the ARM side wrappers for the algorithms implemented. The generated objects 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 mouse are connected to the system for user input.

7. Results and Analysis

The Distances measures like Bhattacharya, Chi-square, Co-relation and Intersection were found to be effective by the accuracy analysis.

The execution times for the algorithms on BeagleBoard-xM (without and with the DSP) are tabulated in Table 1. The performance is 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.

Table 1: Comparison of execution time per frame with and without using DSP

Distance Measure	Time(ms) Without DSP	Time(ms) With DSP
Bhattacharya	40	12
Chi-square	33	9
Correlation	29	9
Intersection	28	9

7.1 Accuracy methods

To determine the accuracy of the proposed approaches to be practically useful, a thorough analysis on execution time is done. The accuracy of tracking is the degree of closeness of measurements of the match value to that of the actual value. Validation of the algorithm can be achieved by analyzing accuracy methods. The input here will be two matched images (template and matched images) and the output will be the accuracy of the match. A sample video is used for accuracy analysis. Let the reference image be 'A' and the matched image be 'B'. Following accuracy measurements are used.

7.1.1 Pixel Cross-Correlation

Steps:

i. For each position in A, find the sum of square difference between pixel values.

$$D = \sum (A(i,j) - B(i,j))^2 \quad (7)$$

ii. Find the percentage of difference in pixel value.

$$P = \frac{D}{(\sqrt{\sum A(i,j)}) * (\sqrt{\sum B(i,j)})} \quad (8)$$

iii. Find accuracy using the equation

$$\text{Accuracy} = 100 - P \quad (9)$$

7.1.2 Pixel Difference Matching

In this method, sum of difference between each pixel is used to find the accuracy.
Steps:

i. For each position (i, j), find the difference in pixel value between the reference image and the matched image.

$$\text{abs}\,(A_{(i,j)} - B_{(i,j)})$$

ii. Find the sum of pixel difference.

$$\text{Sum} = \sum_{a,b} \text{abs}\,(A - B) \quad (10)$$

iii. Accuracy is found by subtracting the sum of difference by the total number of possible values.

$$\text{Accuracy} = \frac{(255N - \text{Sum}) * 100}{255N} \quad (11)$$

Where, N is the number of pixels.

The accuracy of the tracking using different accuracy measures before and after introducing Local Search scheme [6] in the Particle filter is depicted in Table 2 and Table 3. It is clearly visible from the table that the accuracy significantly increases with very negligible cost of computation time after the introduction of Local Search scheme [6].

Table 2: Accuracy results for Particle filter without Local Search Scheme.

Distance Measure	Pixel Cross Correlation	Pixel Difference	Execution time(ms)
Bhattacharya	95.00865	95.99864	12
Correlation	94.08234	96.39664	9
Chi-square	94.09087	97.00675	9
Intersection	93.00986	95.89765	9

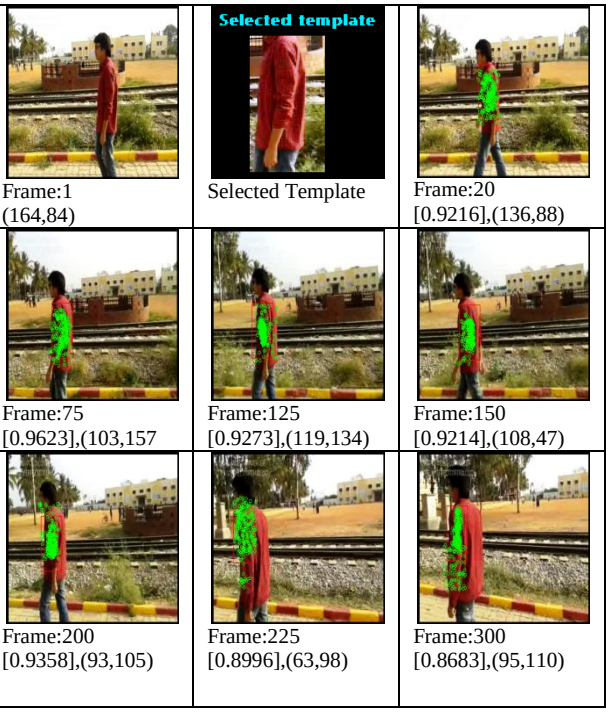
Table 3: Accuracy Results for Particle Filter with Local Search Scheme

Distance Measure	Pixel Cross Correlation	Pixel Difference	Execution time(ms)
Bhattacharya	97.92356	99.86574	14
Correlation	97.87594	98.00078	10
Chi-square	96.99897	97.76589	10
Intersection	97.07897	99.00675	11

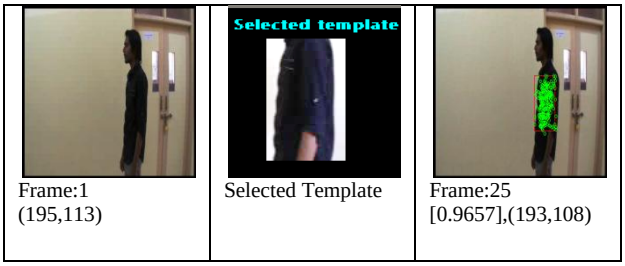
7.2 Results

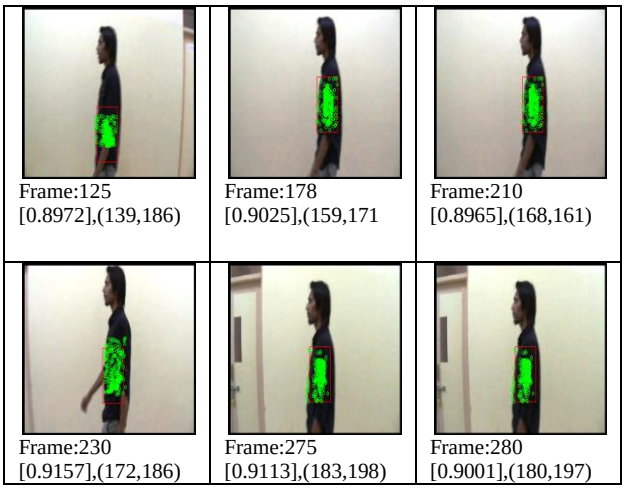
Frame size: 320x240
Template size: 40x80
Number of particles: 250

Outdoor Scene:

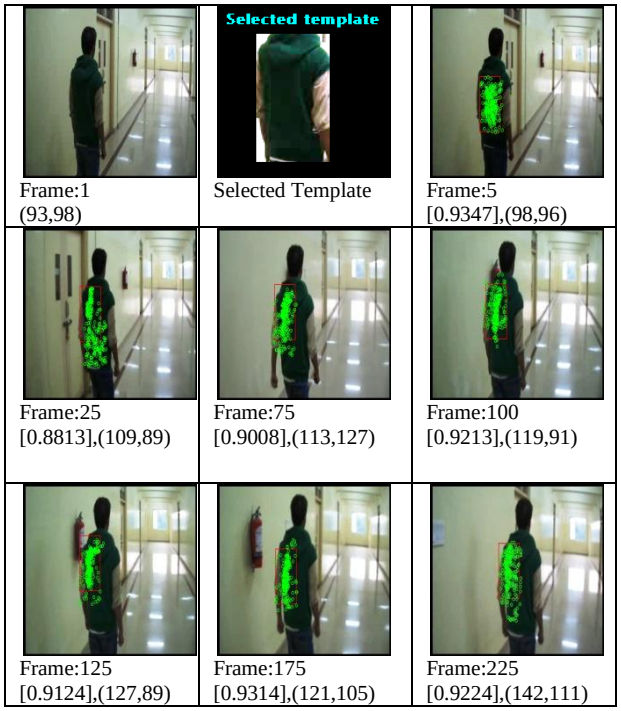


Indoor Scene1:





Indoor Scene2:



Note: values in the large braces represent match value and the small braces represent the match co-ordinates.

References

[1] BeagleBoard-xM Rev C System Reference Manual http://www.beagleboard.org
[2] GStreamer Manual – http://www.gstreamer.net/
[3] Simple Direct media Layer (SDL) – http://www.libsdl.org/
[4] C6Accel Advanced Users Guide http://processors.wiki.ti.com/index.php?title=C6Accel_Advanced_Users_Guide

[5] Jeroen D. Hol, Thomas B. Schfion, Fredrik Gustafsson, "On Resampling Algorithms for Particle Filters", IEEE Workshop on Nonlinear Statistical Signal Processing, 2006.
[6] Juan Jose Pantrigo, Antonio S. Montemayor, Angel Sanchez, "Local Search Particle Filter applied to Human-Computer Interaction"
[7] Katja Nummiaro, Esther Koller-Meier, Luc Van Gool, "An adaptive color-based particle filter"

Dr. Jharna Majumdar obtained BTech in ECE and DIIT in Computer Technology from IIT, Kharagpur in 1969 and 1970, respectively. She Received PhD in Electrical Engineering in 1980. During 1983-89 she worked as a Research Scientist at the Institute for Real Time Computer Systems and Robotics, Karlsruhe, Germany. Currently, she is working as Dean R & D and Prof. & Head of Computer Science and Engineering (PG) at NITTE Meenakshi Institute of Technology, Bangalore. Prior to this Dr. Majumdar served Aeronautical Development Establishment, Defence Research and Development Organization (DRDO), Ministry of Defence, Govt. of India from 1990 to 2007 as Research Scientist and Head of Aerial Image Exploitation Division, Bangalore. Dr. Majumdar has 37 years of experience in R & D and Academics in the country and abroad. She has published large number of papers in National, International Conferences and Journals. Her research areas include Image and Video Processing for defense and non defense application, Robot Vision, Vision based autonomous guided systems, development of Computer Vision Algorithms in FPGA etc.

Parashar Dhakal is a final year undergraduate student from Department of Electronics and Communication at Nitte Meenakshi Institute of Technology (NMIT), Bangalore, India.He joined "Center for Robotics Research", NMIT in 2012 and since then he has been working closely with Dr. Jharna Majumdar on externally funded projects from the Department of Science and Technology, New Delhi, India.His research areas include implementation of image and computer vision algorithms on embedded boards, Embedded System, Computer Networks and Robotics.

Amar Mani Aryal is a final year undergraduate student from Department of Electronics and Communication at Nitte Meenakshi Institute of Technology (NMIT), Bangalore, India. He joined "Center for Robotics Research", NMIT in 2012 and since then he has been working closely with Dr. Jharna Majumdar on externally funded projects from the Department of Science and Technology, New Delhi, India. His research areas include Computer Vision, Embedded System, DSP, Computer Networks and Robotics.

Nabin Sharma Rijal is a final year undergraduate student from Department of Electronics and Communication at Nitte Meenakshi Institute of Technology (NMIT), Bangalore, India. He joined "Center for Robotics Research", NMIT in 2012 and since then he has been working closely with Dr. Jharna Majumdar on externally funded projects from the Department of Science and Technology, New Delhi, India. His research areas include Computer Vision, Embedded System, Computer Networks and Robotics.

Nilesh Kumar Mishra is a final year undergraduate student from Department of Electronics and Communication at Nitte Meenakshi Institute of Technology (NMIT), Bangalore, India. He joined "Center for Robotics Research", NMIT in 2012 and since then he has been working closely with Dr. Jharna Majumdar on externally funded projects from the Department of Science and Technology, New Delhi, India. His research areas include Computer Vision, Microprocessor, Embedded System and Robotics.