Journal of Theoretical and Applied Information Technology
31st August 2017. Vol.95. No.16

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

PARALLEL IMPLEMENTATION OF COVARIANCE TRACKING WITH EFFICIENT MODEL UPDATE

[1]ANUJA KUMAR ACHARYA, [2]BISWA RANJAN SWAIN, [3]BISWAJIT SAHOO
[1,3]Faculty, School Of Computer Engineering, KIIT University, India
[2]Faculty , Trident Academy of Technology, India
E-mail: [1]anujafcs@kiit.ac.in , [2] biswa_gate@yahoo.co.in , [3] bsahoofcs@kiit.ac.in

ABSTRACT

Most of the appearance based tracking algorithm developed in the recent past are characterized by high computation-intensive operations and demands high memory and performance requirements. These appearance based model are also highly sensible to the variation of extrinsic and intern-sic parameter of the feature. In order to track the object under the variation of intern-sic and extern-sic factor , a new model update approach is developed and implemented using a thread level parallelism. Furthermore Particle filter is added to this current method to better handle the back ground clutter, as well as the temporary occlusion. Parallelized implementation achieves significant speedup, and meets the target frame rate under various configurations. Simulation shows that the current parallel method is robust and very effective for the object tracking.

Keywords: *Covariance Matrix, Feature Vector, Thread, Spatial Feature.*

1. INTRODUCTION

All the vision tracking algorithm uses heavy computation and memory intensive operation. Most of these application can be exploited to data level parallelism to enhance the speed of processing. Exploiting the parallelism properly is not an easy task , however, since there are significant memory operations involved in the data operations on image frames. So for such kind of task, parallelization of data are essential in gaining the performance over the serial operation.

In all serial implementation of vision tracking method, image reading operation is carried out serially along with all its sequential operations for processing the frame. In contrast to this, in parallel processing each processor provided with set of operation act on the portion of image data that belongs to its local memory. As the number of processors is increased, more parallelism in the operations can be exploited, but at the same time, more memory is required to store the local sets of image data for the processors. Thus, for optimized implementation of such applications, we need to understand their high-level structure in relation to the target architecture, and we need to balance the trade-offs suitably across the available processors.

Many different types of methodology based on covariance matrix of the image feature has been developed in the recent past for vision tracking [2,3,4] ,Appearance based modeling methods using covariance method for object tracking treated as the best method because of searching the object subspace in the learned space. It's the feature that is the deciding factor in searching the object in the region of interest. Appearance based modeling map the image feature into a fixed size window. Appearance based modeling using the particle filter for tracking the object mainly depends upon the construction of the feature vector by assembling the spatial and temporal feature of the object. Although covariance methods are suitable for real time tracking but some time object drift from the original trajectory because these methods involved with complex computational updation logic and this will go high with the addition of objects in the successive frame of the video.

In summary, To overcome the above cited issues, we proposed a two stage tracking frame work. In the 1st stage uses the parallel method to find the state of the object by obtaining the maximum posterior estimation for each particle and followed to it, In the 2nd stage a simple update strategy is

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

used to adopt the variation of the object to the external variation.

Our work is organized as follows. Section II discus on the related work of the parallel implementation of object tracking. Followed to this, section III discuss the frame work of the proposed model and its updated strategy. In the final Section IV, discuss the performance analysis of our proposed model.

2.RELATED WORK

In Most of the vision tracking application Computation time for finding the region of interest has been a challenging task ,for several year. Exploring parallelized implementations for performance improvement of computationally-intensive operations has been an active research field. In the context to parallelization of task, many image processing application have been implemented on parallel architecture [13]. In [6] Jain etal. presented a parallel implementation of numerical and image processing application on multiprocessor system. The particle filter was introduced to the computer vision community by Isard and Blake in their seminal work [9] in which they presented the CONDENSATION algorithm, which tracks objects based on their contours. This work is further extended by Medeiros & Kleihorst[7] by implementing the parallel approach using color based particle filtering method for tracking the object.The idea of tracking objects based on their color histograms using the particle filter was suggested by Nummiaro et al. [10] and by Pérez et al. [11] around the same time. In spite of a few minor differences, both works present essentially the same algorithm wherein the measurement likelihood is based on the Bhattacharya distance between the current color histograms and the reference color histogram, and the target dynamics are represented by a constant velocity model perturbed by Gaussian noise. An Open MP based parallel architecture is designed by the S Saha et al.[1] for 3D facial pose tracking. Using the course grain data flow model, parallel implementation of particle filter is used in tracking the 3 dimensional face.

Bera et al. [7] reported a real time algorithm based on mean-shift and particle trackers to track pedestrians in crowded scenes at real time rates on a multi-core desktop. Their algorithm was tested with a multi-core processor (4 Cores) and compared. Mohamed Elbahri etal [8] developed a sparse representation based multi object tracking algorithm. An on line dictionary learning is used for object recognition and object detection are classified and indexed according to sparse resolution.

In general, parallel implementation is only suitable to those application that uses the intensive computation in respect to consuming more memory. Implementation of thread level parallelism using OpenMP [12] is an emerging area of parallel programming on shared-memory architectures. It is the first successful effort to standardize shared memory programming directives and is gaining more popularity compared to the message passing model, because of its ease of use. However, for distributed systems, MPI remains a more attractive choice. A combination of MPI and OpenMP at different grains of parallelism is a useful option for many applications. A comparative study of the application of these two paradigms, either jointly or singly, to a multitude of applications is presented in [13]. A variety of image processing applications have been implemented on parallel architectures.

2. FRAME WORK FOR Covariance TRACKING

In the current frame work, we consider the covariance based feature descriptor for representing the referential model. Similarity comparison between the two covariance descriptor can be evaluated using the Riemannian geometry based Fostner distance. The tracking explored in work of [14] is combined with the particle filter and then implemented on the Open MP platform to parallel execute the thread. As shown in the figure 1, Our parallel framework consists of two phase. In the 1st phase, a low dimensional feature vector is constructed ,distribution of particles in the search space and the likelihood estimation is evaluated. Followed to this in the 2nd phase focus on the construction parallel thread for the independent particles ,re sampling and the efficient model update is carried out.

3.1 Covariance Matrix Representation of Image

Using Tuzzel et al[2], Covariance matrix representation, The observed Image I of MXN one dimensional or three dimensional color image is converted into d dimensional feature image of size M X N X d. Mathematically this mapping can be written as

$$F(n,p)= Ø(I,r,c) \text{ ----------(1)}$$

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Here the function Ø is a mapping function and it

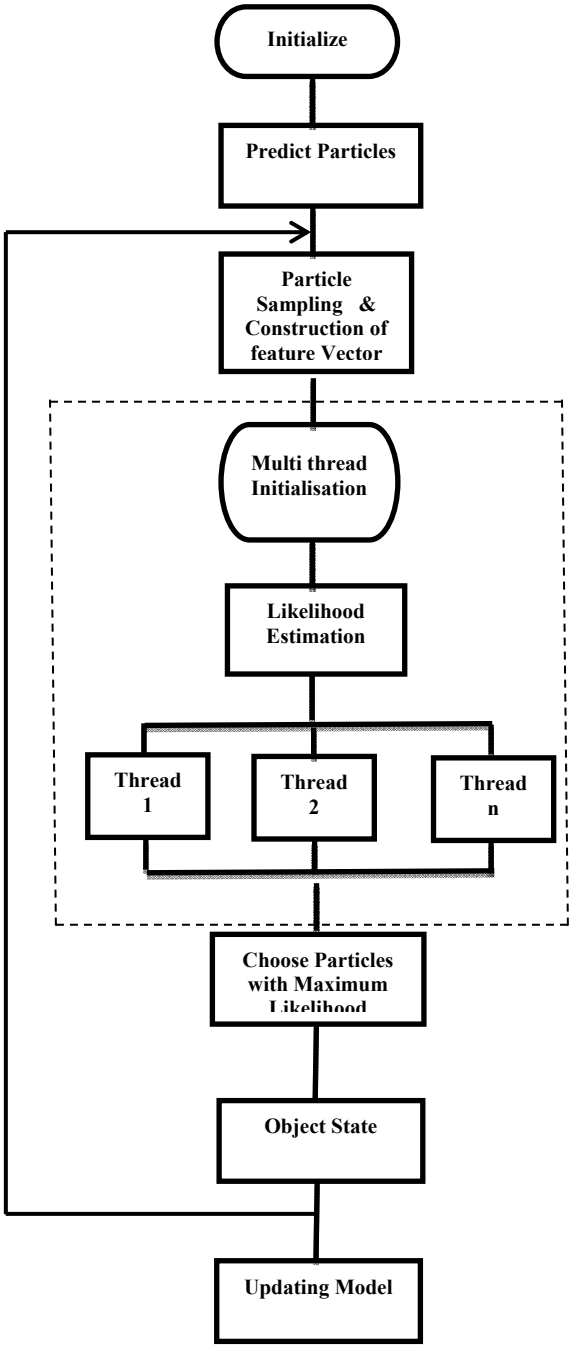

Figure 1: Proposed Object Tracking Model

can be any mapped such as color , image gradient or edge magnitude. For a given rectangular region R **C** I ,denote the {fi}i=1....n as the d dimensional feature point obtained by Ø within R. In this paper each pixel of the object is defined through the set of feature as

$$f_k = [x\ y\ I(x,y) I_x I_y \sqrt{I_x^2+I_y^2}.\arctan\frac{I_x}{I_y}] \quad \text{.....(2)}$$

where x,y, are the spatial coordinate and I(x,y),Ix,Iy are the intensity value and its deviation of the pixel along x and y. Consequently, the image region R can be represented as *dXd* covariance matrix .

$$S_r = \frac{1}{(mn-1)}\sum_{i=1}^{n}(y_{ij}-\hat{y}_i)(y_{ij}-\hat{y}_i) \text{ - - - - - (3)}$$

The covariance matrix descriptor of a gray scale or color image region is a 7 X 7 or 17 X 17 symmetric matrix.

3.2 Target Searching:

For the practical implementation, each pixel is represented with *d* dimensional feature surrounding object region of the image having M x N dimension. Those feature vector again represented into the compact covariance structure. A Fostner based distance is employed in the covariance based structure for matching the best candidate in the target frame of the next frame.

$$\rho(C_i,C_j) = \sum_{k=1}^{d}\left(\sqrt{\log^2\lambda_k(C_i,C_j)}\right) \text{.....(4)}$$

where $\lambda_k(C_i,C_j)$ are the generalized eigenvalues of (C_i,C_j).

3.3 Bayesian State Inference

Predict the future location and estimate the current state from the all previous observation is the main aim of using the Bayesian statistics . The set of observed state is represented as the $Y_{1:t} = y_1, y_2, \ldots, y_t$.The optimal state X_t of the tracked object can be found by finding the maximum posterior estimation .The density propagation for the tracker can be formulated as

$$p(x_t \mid y_{0:t}) \propto p(y_t \mid x_t)\int p(x_t \mid s_{t-1})p(x_{t-1} \mid y_{0:t-1})dx_{t-1}$$

Using Particle filtering [9], The posterior estimation $p(x_t \mid y_{1:t})$ can be approximated by a finite set of N_s samples or particles the particle $[x_t^i]_{i=1}^{N_s}$ with importance weights $[w_t]_{i=1}^{N_s}$ The particles sample x_t drawn from the an importance

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

distribution $q(x_t \mid x_{1:t-1}, y_{1:t})$ which for simplicity set to the dynamic model. We apply an an affine image wrap to model the object motion between the two consecutive frame. The state transition distribution is modeled by the Brownian motion I.e $p(x_t \mid x_{t-1}) = N(x_t; x_{t-1}, E)$ where E is the diagonal covariance matrix whose diagonal elements are corresponding variance of the respective parameters.

The feature vector is extracted for each particle and then the particle that yields the best likelihood value is said to give the position of the object in the frame. An exponential function of the distance is adopted as the local likelihood in the particle filter.

$$p(y_t \mid x_t) \propto \exp(\lambda \rho^2(C^0, C(x_t^n)))$$

The similarity measurement of two covariance matrix can be found out using the equation (4).

3.4 Resampling :

Resampling is a critical operation in particle filtering because with time, a small number of weights dominate the remaining weights, thereby leading to poor approximation of the posterior density and consequently to inferior estimates. This portion of particle filer can't be carried out in parallel because of data dependencies of the particles. In the proposed work the resampling part of the particle filter is evaluated in serial execution order. Once the likelihood estimation of all the particles are completed, then they are assigned to array to keep this value preserved. Using the Systematic Sampling approach, each of the normalized valued of the particle are processed in an array to find the cumulative sum of the weights and then compare it against the set of random number to decide which particle has the maximum chance of matching. So for each iteration, following this resampling approach reduce the weight degeneracy problem.

3.5 Model Update:

As the covariance tracking in appearance based modeling is most sensible to the external variation of the feature, In order to reduce this false detection of the object in the target frame ,Model update step is required. In the current paper we have devised a variance based update logic for the model update. With each of the detected covariance matrix, referential model is updated by taking the average of the detected covariance matrix with with the existing covariance matrix of the referential region.

The average covariance of two multivariate sample S1,S2 having the covariance matrix C1 and C2 can be determined as

$$C_1 = \begin{pmatrix} \sigma_{1,11}^2 \sigma_{1,12}^2 \sigma_{1,13}^2 \cdots \\ \sigma_{1,21}^2 \sigma_{1,22}^2 \sigma_{1,23}^2 \cdots \\ \sigma_{1,31}^2 \sigma_{1,32}^2 \sigma_{1,33}^2 \cdots \end{pmatrix} C_2 = \begin{pmatrix} \sigma_{2,11}^2 \sigma_{2,12}^2 \sigma_{2,13}^2 \cdots \\ \sigma_{2,21}^2 \sigma_{2,22}^2 \sigma_{2,23}^2 \cdots \\ \sigma_{2,31}^2 \sigma_{2,32}^2 \sigma_{2,33}^2 \cdots \end{pmatrix}$$

$$\sigma_{1,ij}^2 = \frac{1}{(m)} \sum_{i=1}^{m} (x_i - \mu_{1i})(x_j - \mu_{1j}) \text{ ----- (3)}$$

$$\sigma_{2,ij}^2 = \frac{1}{(n)} \sum_{i=1}^{n} (x_i - \mu_{2i})(x_j - \mu_{2j}) \text{ ----- (4)}$$

The average covariance structure of this two sample can can be obtained

$$\sigma_{o,jk}^2 = \frac{1}{(m+n)} \left[\begin{matrix} m\sigma_{1,jk}^2 + + n\sigma_{2,jk}^2 + \\ m(\delta_{1j}\delta_{1k}) + n(\delta_{2j}\delta_{2k}) \end{matrix} \right] \text{ ..... (5)}$$

If n=m then

$\delta_{1j} = \delta_{2j} = \delta_j$ and $\delta_{1k} = \delta_{2k} = \delta_k$

and equation become

$$\sigma_{o,jk}^2 = \frac{1}{(2)} \left[\sigma_{1,jk}^2 + \sigma_{2,jk}^2 + 2(\delta_j \delta_k)) \right] \text{ ..... (6)}$$

4. PARALLEL COMPUTATION

The portion of the graph marked with dotted lines shows a group of operations that occur N times for a given image frame, where N is the total number of particles of the system. These N executions are independent of each other as there are no data dependencies among them. As shown in the figure 1 if we consider the marked portion of the graph executed by one single thread then the computation time must be greater than if it is executed by the n number of parallel thread. In the present work the execution of the parallel thread is carried out using the openMP platform.

4.1 OPENMP Execution Model :

The OpenMP API is a fork-join model for the parallel execution of application program. Multiple

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

threads of execution perform tasks defined implicitly or explicitly by OpenMP directives. OpenMP is intended to support programs that will execute correctly both as parallel programs (multiple threads of execution and a full OpenMP support library) and as sequential programs. However, it is possible and permitted to develop a program that executes correctly as a parallel program but not as a sequential program, or that produces different results when executed as a parallel program compared to when it is executed as a sequential program.

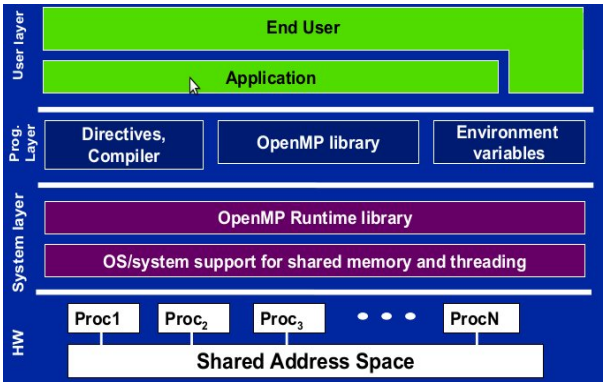

Figure 2: Open Mp Layout

As shown in the figure 2, OpenMP library and the callable runtime library routine that extends c++ to exploit the shared memory parallelism. In the shared address model every processor has direct access to every other processors in the system.

An OpenMP program begins as a single thread of execution, called the initial thread. The initial thread executes sequentially, as if enclosed in an implicit task region, called the initial task region, that is defined by an implicit inactive parallel region surrounding the whole program. In the present case we have started with the single thread that contains the referential model and the remaining n-1 thread is created for estimating the likelihood of the n-1 particles in the target frame of the video. Each thread undergoes for the computation of similarity measurement for finding the local likelihood of the particle as $p(y_t \mid x_t) \propto \exp(\lambda\rho^2(C^0, C(x_t^n)))$.

In the next section, we have focus briefly the algorithm for the parallel implementation of our method along with the pseudocode.

4.2 Algorithm For the Parallel Implementation

Step 1: Initialize the $C = C1, \mu_1, \sigma_1$ of the referential model.

Step2 : Compute the Covariance $C_{i=1\ldots n}$ of n number of particles in parallel.

Step 3. Calculate the likelihood of each particle as

$$p(y_t \mid x_t) \propto \exp(\lambda\rho^2(C^0, C(x_t^n)))$$

Step 4: Normalize the probability distribution of particle

$$p(y_t^n) = P(y_t^i) \Big/ \sum_{i=1}^{n} p_{y^i}$$

Step 5: using the equation (x)

$$\sigma_{0,ij}^2 = \frac{(\sigma_{ii}^2 + \sigma_{jj}^2)}{2} + \delta_i\delta_j + \frac{\delta_i(\sigma_{ii} - \sigma_{jj}) + \delta_j(\sigma_{ii} - \sigma_{jj})}{2\sqrt{(n)}}$$

each term of the referential covariance model can be updated.

The serial version of this algorithm can be parallelism using the parallel thread. To evaluate the likelihood estimation of each particle., it depends on the estimating the covariance matrix of that region that in terms required to read the area of interest of the image from the respective frame and store it into the local memory.

The serial version of this algorithm can be parallelism using the parallel thread. To evaluate the likelihood estimation of each practice it depends on the estimating the covariance matrix of that region that in terms required to read the area of interest of the image from the respective frame and store it into the local memory. As shown below the table the #pragma omp create n number of parallel thread at run time. Each thread then undertakes the responsibility in executing the likelihood of the particle.

Psudo code for the n numbers of Particle

```
object_parallel_read(I,x1,y1)
{
x=imread(I);
    for (i=y1 ,i<= y1 + M ;i++ )
        for (j= x1 j<=x1 + N ;j++)
      //Construction of the feature matrix
```

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 1

Particles/Threads	40 Particles	80 Particles	160 Particles	200 Particles
1	20	30	40	70
2	15	25	28	48
4	6	8	12	20
8	4	6	7	10
16	6	10	13	23

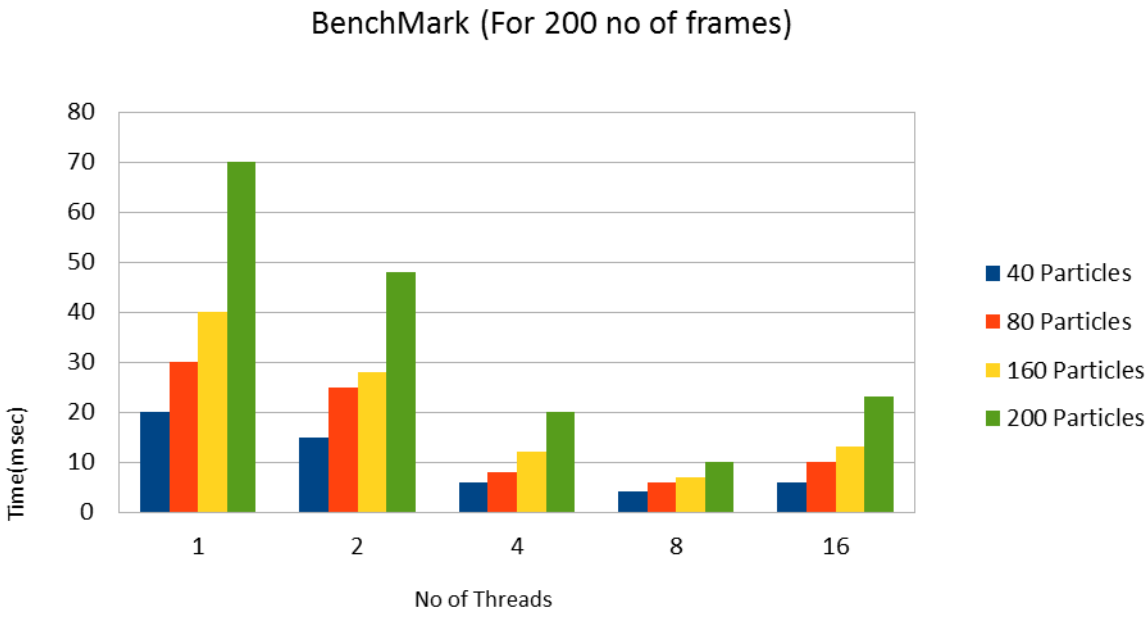

Figure 3 Speed Up Measurements For Different Number Of Thread

```
Feature_matrix[i1++][1]=j;

        Feature_matrix[i1++][2]=i;
         . .....
         .....
         .....

         Feature_matrix[i1++][d];
    end


 k=k+1;
   end
 Cov_feature_Mat =
cov(Feature_matrix,M,N,d);
}
```

```
/* Fork a team of threads giving them their own
copies of variables */


#pragma omp parallel  num_threads(n)
{

id=omp_get_thread_num();
paralle_object_read(id,a);
particle_prediction(n);
}
```

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

5. PERFORMANCE EVALUATION

Our Proposed Model is implemented using opencv -2.4.2 libraries on Ubuntu platform and this simulator is configured with OpenMp to support the thread level parallelism. All the experiment are conducted in a intel core i-7 machine. Both serial as well as the parallel version of the algorithm are tested by executing maximum 8 thread at a time. The covariance matrix descriptor of a color image region is represented by 23 X 23 symmetric matrix.

Pipelined implementation is followed in constructing the parallel threaded approach of Our Model. The main thread starts reading an image and after that n numbers of parallel thread are created using # pragma omp. These parallel thread construct the feature vector and then find the posterior distribution of the particle. The vital part of particle filtering are the predicting the particle for the next frame ,the weight estimation of particle and the re sampling the particle. Using the gausian distribution, state prediction of particles are obtained and this also requires the generation of the random numbers that is distributed normally across the search space. Various types of experiment are conducted and applied on two vision dataset by taking different number of thread on a Intel(R) Core(TM) i7 CPU 965 @ 3.20GHz machine. The Performance measurement parameter speed of computation and scalability are evaluated below for our parallel approach.

Speed Up:

Speed up is the ratio between the execution time of a serial process to the parallel process. It is practically impossible to obtain the n times speed up rate by using the n number of thread. For parallel computing, Amdahl's law states that if F is the fraction of a calculation that is sequential (i.e. cannot benefit from parallelization), and (1 – F) is the fraction that can be parallelized, then the maximum speedup that can be achieved by using N processors is

$$SpeedUp = \frac{1}{F + \frac{(1-F)}{N}}$$

The performance results shown in Figures 3 and Table I are obtained for numbers of particles varying from 40 to 200 and numbers of threads varying from 1 to 16. When only one thread is used, essentially the serial implementation is realized and hence the serial performance is given by the execution times using one thread. As may be observed from the results, the implementation using threads in OpenMP shows very large performance improvements compared to the serial version (one thread).

It was observed that the execution times did not improve by merely increasing the number of threads. In fact, there was generally a point beyond which we found that increasing the number of threads resulted in the execution time to increase. The best performance is obtained for 8 threads and not the maximum number of threads, which was 16. Beyond 8 threads, the amount of time spent in scheduling and coordinating the threads starts overshadowing the gains obtained from parallelization, and hence the execution time starts increasing.

Scalability

Scalability parameter is a widely used parameter for testing the performance of parallel system for larger problem size. For scalability evaluation, the multi-core frameworks should be tested in different problem sizes by testing the frameworks using different frames. In this paper, the same object tracking algorithm was considered for 100, 200, 400 and 800 frames. The Scalability testing experiment is carried out on the two vision dataset of moving person where the frames are captured from a stationary camera. As shown in the figure 4,our parallel algorithm is quite capable of tracking the object for longer sequence of video frame.

6. CONCLUSION:

Parallel implementation of Appearance based particle filtering algorithm is very effective in tracking the non-rigid object. It is not only reduce the computation time but also our current approach is very much effective in updating the model to external variation of the object. One of the limitation of our current approach is the necessity to store all the detected feature vector. As the construction of feature vector is takes the larger amount of memory and the scalability of the system will be degraded due to this memory contention issues. This problem can also be irradiated if the reference model is updated with each of the detected frame. Thread level parallelism can be further be extended to process level parallelism for the system that uses tracking the object using multiple camera connected to client system. This can be an area that can be extended to achieve the robust and accuracy of the algorithm.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Figure 4 : Tracking The Moving Person For The Sequence Of Video Frame

REFERENCES

[1] Sankalita Saha, Chung-Ching Shen, Chia-Jui Hsu, Gaurav Aggarwal, Ashok Veeraraghavan, Alan Sussman and Shuvra S. Bhattacharyya, "Model-Based OpenMP Implementation of a 3D Facial Pose Tracking System . " International Conference on Parallel Processing Workshops (ICPPW'06) .

[2]Fatih Porikli, Oncel Tuzel " Covariance Tracking using Model Update Based on Lie Algebra" Proceedings of the

[3]Y. Li, "On Incremental and Robust Subspace Learning," Pattern Recognition, vol. 37, no. 7, pp. 1509-1518, 2004.

[4] D. Skocaj and A. Leonardis, "Weighted and Robust Incremental Method for Subspace Learning," Proc. Ninth IEEE Int'l Conf Computer Vision, vol. 2, pp. 1494-1501, Oct. 2003.

[5] T. Yu and Y. Wu, "Differential Tracking based on Spatial-Appearance Model(SAM)," in Proc. CVPR, Vol. 1, pp.720-727, June 2006.

[6] B.M. Maxiarz and V. K. Jain. Rapid Prototyping of Parallel Processing Systems on TESH network. In Proceedings of Ninth International Workshop on rapid System Prototyping, 1998.

[7] Henry Medeiros, Xinting Gao Johnny Park , Richard Kleihorst , Avinash Kak "A Parallel Implementation of the Color-Based Particle Filter for Object Tracking "2010

[8]Mohamed ELBAHRI ,A Novel Object Position Coding for Multi-Object Tracking using Sparse Representation ,I.J. Image, Graphics and Signal Processing, 2015, 8, 1-12

[9]I SARD , M., AND B LAKE , A. Condensation – conditional density propagation for visual tracking. International Journal of Computer Vision 29, 1 (1998), 5–28.

[10]N UMMIARO , K., KOLLER -M EIER , E., AND G OOL , L. V. A Color-Based Particle Filter. In First International Workshop on Generative-Model-Based Vision (2002).

[11] P ÉREZ , P., H UE , C., V ERMAAK , J., AND G ANGNET, M. Color-based probabilistic tracking. In ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part I (London, UK, 2002), Springer-Verlag, pp. 661–675.

[12]L. Dagum and R. Menon. OpenMP: An Industry-StandardAPI for SharedMemory Programming.In IEEE Computational Science and Engineering, 5(1), 1998, pp. 46-55.

[13]S. Bova, C. Breshears, H. Gabb, R. Eigenmann, G. Gaertner, B. Kuhn, B. Magro, S. Salvini and H. Scott. Parallel Programming with Message Passing and Directives. Computing in Science & Engineering, 3(5), 2001, pp. 22-37.

[14] AK Acharya, B Sahoo , "Visual Tracking on Riemannian Space Using Updated Standard Deviation Based Model ", International Journal of Computer Vision and Signal Processing, 5(1), 31-37(2015) ,ISSN: 2186-1390 , IJCVSP, CNSER.