Proceedings of the International Conference , "Computational Systems for Health & Sustainability"
17-18, April, 2015 - by R.V.College of Engineering,
Bangalore,Karnataka,PIN-560059

Processing videos using parallel computing: A Novel Approach

Mahesh B Fattepur
M.Tech CSE – II sem
R.V College of Engineering Bangalore-560059, India

Jayashree B.Huttanagoudar
M.Tech CSE – II sem
R.V College of Engineering Bangalore-560059, India

Abstract— In this paper, the proposed framework is presented that supports acquiring high-resolution video's from the low-resolution. The high-resolution videos could be used in tagging, identifying and tracking people. We concentrate on two aspects. One, data simplification method as the algorithm required for conversion large amount of data processing which is run in parallel. Second, is building a parallel video processing techniques as pipeline for analyzing image modules such as face detection, recognition and tracking so that multiple people can be identified more efficiently and smoothly with increased performance and computational efficiency. Parallel processing techniques makes the use of super resolution algorithm obsolete for major modification in generating high-resolution video images. Recognition of multiple people with super resolution can be tracked from real time live videos or it could be recorded one.

Keywords-Video analysis, human identification, face detection/tracking, video parallel computing/processing, high/super video resolution**.**

I. INTRODUCTION

The use of surveillance cameras in work places, hospitals, airports, educational institutes, data centers and now a day is used in residential places which provide a large amount of data to analytics. These all aspects are leading its importance in the fields of securities, education, health care, smart villages and smart home and in many other places. This job aimed in analyzing uncontrolled captured surveillance videos and building a human-centric system that identifies, tracks people in videos when viewed in high resolution video.

Many studies on object tracking in moving real time videos are proposed that are still having the significant challenges in identifying, recognizing, tracking people with low resolution capture because of high disturbance or variations and cluttered background. There is a large amount of computations required for tracking, identifying people in live videos, even requires a large amount of data processing in conversion algorithm for getting high resolution videos.

Basically, the two aspects of issues have to be resolved. One, techniques for increasing the spatial resolution of videos that involves the resizing of independent frames in videos to higher spatial dimensions with digital signal processing algorithms [1]. It sometimes results in blurring, aliasing when interpolation techniques are used. The high-resolution algorithms require a large computational data and are expensive.

In this paper, we concentrate on using desired frames for processing as an input to algorithm so that the video can be viewed in super-resolution with less computational expensive. We take the subsets of the video frames pixels that are captured from surveillance to be processed by the algorithm in parallel. Second, pipeline processing techniques that uses multiple stages for processing selective video frames pixels with multiple threads to be used in parallel for face detection, face recognition and face or head tracking which are robust to human actions with fast implementations.

The above survey can be used to prove that the parallel processing for tracking multiple moving people from selective super resolution algorithm could be efficient and less expensive with the ability of live detection, tagging, identification and tracking of multiple people.

The paper is described as follows. Section II describes the proposed framework for data simplification; recent approaches on video face recognition and tracking are introduced, Section III The parallel pipeline processing method, Section IV provides a demo of existing video processing written in C++ , in Section V we present the conclusion and finally in Section VI with the references.

II. PROPOSED FRAMEWORK

The framework consists of two major parts that is described in this section. The first one, called simplification identifies the data as complex or simple regions. Complex regions are defined with more visually significant information's than the simple regions. The complex computational algorithms are used on complex regions to get the better visual information. Less complex algorithms as interpolations are used in the simple regions. For better efficiency and processing time we go with classification of complex and simple regions. Secondly, the recent approaches on image frame detection.

A. Simplification Approach

Videos carry huge amount of redundant data and the strategies is to reduce the redundant information's using the complex accurate algorithm that reduces the complexities. That is use of complex (and accurate) algorithms. If the selection algorithm is well designed, the videos generated with this method can be hard to distinguish from the signals generated using only the more complex algorithms. The following data simplifications methods can be used for reducing the complexilities: significant information selection, contour-giuded processing and differential encoding.

1) Significant Information Selection (SIS): Typically, each video frame has three color channels they are YUV color. With the naive approach in increasing the frame size of a video that uses super-resolution algorithms in each of the three color channels. To save computational resources, super-resolution algorithms should be applied only to the most important color channel. The SIS simplification consists of using the super-resolution algorithm in the luminance channel and an interpolation algorithm in the two chrominance channels. The quality of outcome generated by interpolation is typically lower than the quality of outcome generated by a super-resolution algorithm.

2) Contour-Guided Processing (CGP): CGP strategy divides the spatial areas of the frames into two types:

1. Edges and high-detail regions and
2. Mostly uniform areas.

The regions that contain significant or detail information are termed as “Regions of Interest” (ROI), and the uniform regions are classified as Secondary regions (Rs). An example of the classification is depicted in Fig. 1 for the video ‘Paris’. Fig. 1-(b) shows the output of the canny algorithm, corresponding to the original video in Fig. 1-(a). In this figure, black areas correspond to Rs and the white areas to the ROI.

For magnification purposes it is necessary to identify the secondary regions as uniform data. To accomplish this, we partition the segmented image into n×n blocks. Then, we verify which blocks contain ROI pixels and then indicate it as complex blocks (Bc) and for the non ROI pixels we indicate as simple block (Bs). With this approach we can apply the magnification algorithm for more detailed information from the complex pixel blocks. Complex blocks are processed using the super- resolution algorithm and simple blocks with an interpolation algorithm. Figs 1-(c) and (d) show the result of the block classification for two different block sizes: 4×4 and 8×8, respectively. The white blocks are Bc blocks (complex information), while the black ones are Bs blocks (simple information).

3) Differential Coding (DC): The CGP and SIS exploit the image in data level that is operating only on the spatial domain. The temporal redundancy which is the information in neighboring frames is very similar.To exploit this redundancy, the consecutive frames difference is calculated and then we use the step 1 and step 2 for identifying the regions of the frame.

Fig. 2 shows an example of using this method for the video ‘Paris.

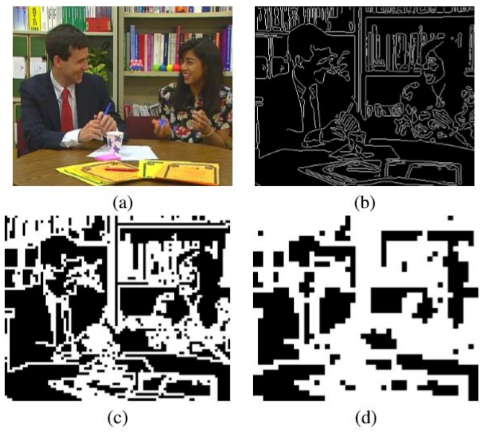

Fig. 1. Results of applying CGP to first frame of video Paris: (a) original, (b) output of Canny algorithm, (c) frame partitioned using 4×4 blocks e (d) frame partitioned using 8×8 blocks. Bc and Bs blocks in (c) and (d) are shown in white and black, respectively.

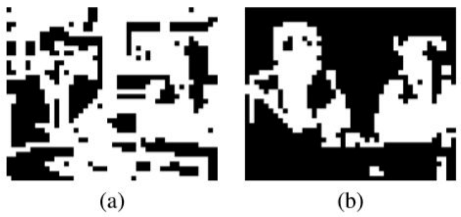

Fig. 2. Results of applying CGP (8×8 blocks) to: (a) the second frame of the video Paris and (b) the difference between the second and the first frames of the video Paris. Selected blocks are shown in white.

B. Video Face Tracking

For object tracking, there are several approaches, including the Kanade-Lucas-Tomasi (KLT) [2], the mean shift [3] and the particle filtering [4] algorithms. Most video face tracking approaches are based on one or more of these basic methods. For example, in [5], the KLT tracker is used to track the ROI called as points of interest throughout video frames, and then each face track is formed by the faces detected in different frames that contributes large enough number of tracked point. The CamShift method is an extension of the mean shift algorithm with an adaptive region-resizing step.

A particle filter based method in [6] that uses an adaptive target model. This approach considers the two constraints to avoid the target drift for computing the particle weights: one is face pose constraint; the other is alignment constraint which determines whether or not the object image contains a well-aligned accurate face that is cropped.

The tracking-learning-detection (TLD) framework is applied to faces in [7] which employ the KLT tracker and an offline trained face detector

C. Video Face Recognition

Face recognition or clustering of candidate pixels from the high-resolution video is different and difficult from the images. Major problem could be utilization of information to detect the face in videos. A method is proposed in [8] for non-supervised automatic labeling of human faces in videos. It uses faces inside the same track as positive examples, and face pairs of different persons appearing in the same video frame as negative examples to learn specific face recognition metrics which are then used to define a distance between face tracks for identification purposes.

The problem of matching high-resolution gallery images with surveillance quality probe videos is addressed in [9].

A system is proposed in [10] that integrates detection, identification and tracking of human targets in video from a single static camera.

From the above details, it can be seen that there is still a big gap in getting real-time multi-face tracking and recognition in high resolution video. Several aspects including designing efficient parallel computing architectures, efficient tracking of multiple object faces, and human recognition utilizing multiple views of the face.

III. PARALLEL VIDEO PROCESSING

A. Distribution and Parallel ProcessingApproachs

The parallelization of the simplification step is simpler. Initially, a buffer is created containing a set of consecutive frames. These buffers contain all the sizes frames of the videos and enough number of these buffers are created for any sized video frames. Depending on the video size and the available computational resources, several buffers may be created in sequence.

These buffers are processed with distributed approach and the enough resources are providing to individual buffers. In the case of multiple buffers for a video all the below steps are processed. The parallel processing steps described for each buffer are simplification, classification, distribution, and reconstruction, as described below.

1) Simplification and Classification: The simplification step uses a set of consecutive frames and the consecutive frames are uniquely numbered according to their time frame positions. These positions are stored in a queue T, which is a shared data structure accessible by all processes. Then according to the available resources the processes are assigned to each data set. Example, if a buffer containing K distinct frames in an environment with K processors, then every distinct frame is assigned to distinct processors that all will be run parallely. Each process is defined with pair as P(t,ei), where t is the frame in the time position and ei is the ith-stage of the simplification step. Besides the selec- tion of significant information, simplification and classification stages includes three more stages: differential coding, contour-guided processing, and block classification. The first stage, e1, consists of computing the differential frames. In this case, during process p(t,e1) the difference between frames t and t+1 is computed, considering the number of reference frames the user wants to keep. Meaning, if there are r referential frames, the algorithm keeps all frames with position equal to:

$$t \bmod (K/r)=0$$

To save the information about the type of frame, a queue B is used that serves as a bitmap, indicating whether the frame is differential or not. In the second stage, e2, the process p(t,e2) detects the ROIs of frame t. Similarly, in the third stage, e3, the process p(t,e3) detects simple and complex regions in the frame, classifying them as selected for the blocks Bc or unselected for the blocks Bs. The stages e1, e2, and e3 are illustrated in Fig. 3.

2) Distribution: The partitioning used in the simplification defines that there is an independence of data in each consecative diferrent video frames. Where all the set of operations are performed on each block independently.

Each block after simplification is encapsulated in data structure knows as cell to distribute data uniformly among the different independent processes. All these cells of different blocks contains the Boolean flags indicating the selected block or unselected block, three integer variables indicating the block and frames pairs and the rest two integers represents the horizontal and vertical spatial position of the block in the frame, and finally a matrix containing the pixel values of the block. The cell contains the class of the encapsulated block and its temporal and spatial information for generating the data distribution with a good load balance, two queues are used: F_{in} and G_{in}. When the true flag is present in encapsulated cell structure, then it is queued in F_{in}. Otherwise, is queued in G_{in}.

Hence forth, these two queues maintains the load balance, all the frames are arranged based on demand basis rather than the frame time positions. Both the F_{in} and G_{in} are shared data structures. Although each queued cell may be accessed by all processes, each process uses its own queue that is not shared. When all the blocks are differentiated and the queues are filled, the stages e1, e2, and e3 are independently executed by the processes.

Figure shows the distribution and classification of the internal and external data structures. For the three frames shown, each highlighted square is a cell. Different colors are used to indicate the frame class and the block type. Red, yellow, and pink regions are formed by Bc blocks of frames t, t +1, and t +2, respectively. Green, blue, and gray regions contain Bs blocks of frames t, t +1, and

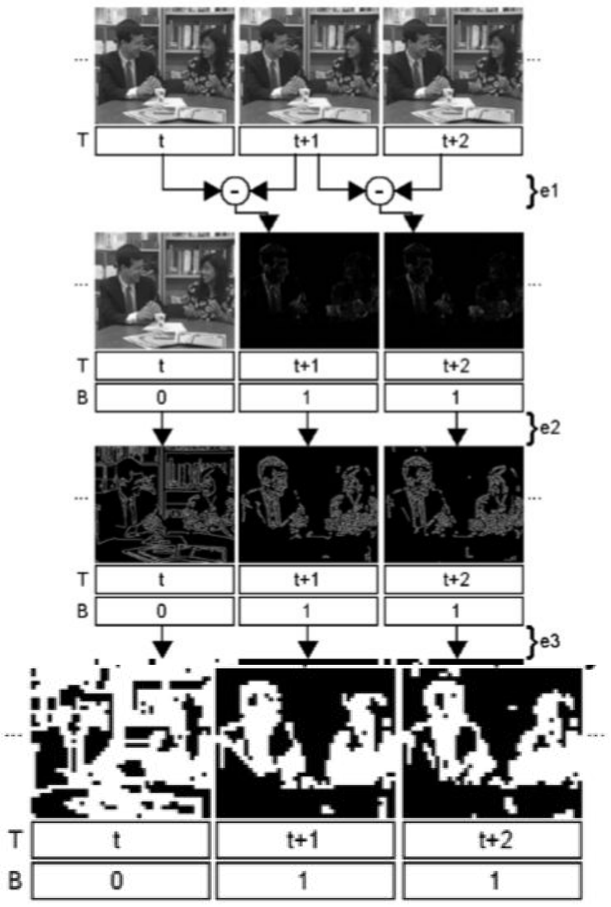

Fig. 3. Parallel processing of simplification and classification steps.

t+2 respectively. When these cells are queued in Fin, the elements may be treated uniformly, allowing this structure to be equally distributed among the processes. The contents of the queues F_{in} and G_{in} are distributed to all the nodes. An environment with K nodes, there will be K meshes. In heterogeneous architecture a set of processors are used in each node. This approach the framework can be executed in a distributed memory system, where each node has a set of processors using shared memory. Fig. 4 shows the distribution and processing over the nodes, which correspond to sending the content of meshes k and k+1 to nodes k and k +1. In this figure, each mesh has only four cells, which are distributed to the four processors contained in each node.

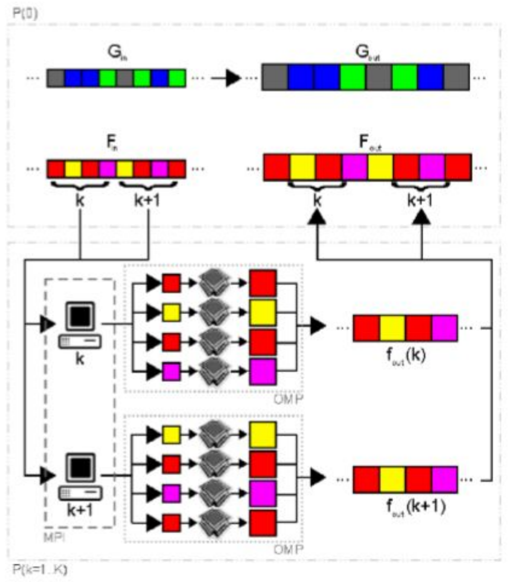

Fig. 4. Distributed processing over the nodes.

3) Reconstruction: The elements are processed in f_{in} and sent to f_{out} where the final elements of this last queue are queued in f_{out} by each node. These final queue is containing the ordered elements as of original positions. The sorting is shown in Fig. 5. First, we iterate through G_{out}, checking the correct position which is encapsulated in each cell structure. Using the position information, we copy the encapsulated block to the output video. Then, we process F_{out} in the same way.

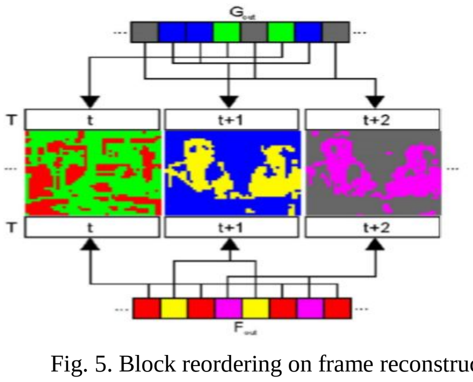

Fig. 5. Block reordering on frame reconstruction.

A serial restoration is shown in Fig. 6 because the frame restored on t +1 depends of t, t +2 depends on t +1, and so on.

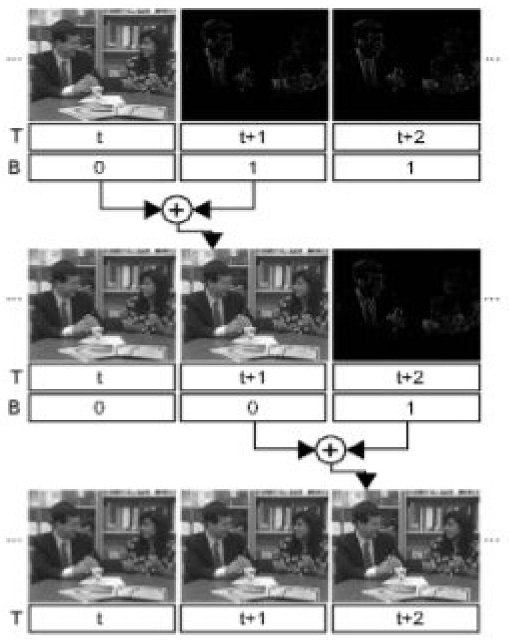

Fig. 6. Reconstruction of differential frames

4) Avoiding the Blocking Effect: The decomposition of the frames in independent blocks generates a blocking effect, which consists of visible edges around the borders of blocks, as illustrated in Fig. 7. To avoid the blocking effect, we simply introduce an edge of zeros which is a padding effetcs before performing resizing super-resolution or interpolation. After the resizing process, the padding is removed from the magnified block.

Fig. 7. Reconstructed frame without padding

B. VIDEO PROCESSING PIPELINE

Once the super resolution video is generated form the above simplification method then the parallel pipelining approach is done to identify the moving objects by integrating the different image analysis modules so that the system can run efficiently and smoothly.

1) The Video Processing Workflow: The proposed video processing workflow is shown in Fig.8 (a). In the software implementation, the main thread includes the blocks of frame acquisition, image preprocessing, face detection and data association, face/head tracking and result display; while child threads are created for the blocks of dynamic face modeling and face recognition. We adopt the Intel TBB which is a popular C++ template library for task parallelism [11]. Fig.8 (b), each row represents the processing of one video frame. That is, the processing of one frame is divided into four stages, and consecutive frames are processed in parallel but at different stages.

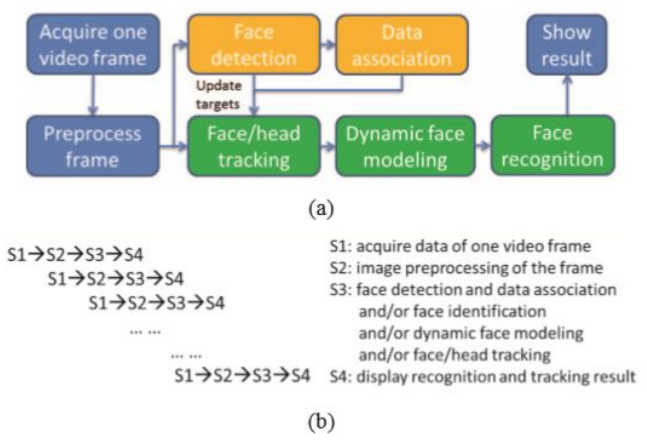
Acquire one video frame
Face detection
Data association
Show result
Update targets
Preprocess frame
Face/head tracking
Dynamic face modeling
Face recognition
(a)
S1→S2→S3→S4
S1→S2→S3→S4
S1→S2→S3→S4
... ...
... ...
S1→S2→S3→S4
S1: acquire data of one video frame
S2: image preprocessing of the frame
S3: face detection and data association
and/or face identification
and/or dynamic face modeling
and/or face/head tracking
S4: display recognition and tracking result
(b)

Fig. 8.The proposed video processing pipeline: (a) the workflow and (b) the assembly line model.

In S1, image data of one video frame is acquired and put to the buffer shared by S1 and S2. In S2, the frame is preprocessed, including normalization, gradient computation and integral image calculation, with results put to the buffer shared by S2 and S3. In S3, there are several major image analysis modules. Face Detection is applied once every N frames, followed by Data Association. The results are used to update the list of tracking targets. Face/Head Tracking is conducted on every frame for objects on the current target list. Dynamic Face Modeling is called once every M frames on some specific faces. And Face Recognition is done on objects that are not yet identified. In S4, the task is to ensure a human recognition and tracking. All the stages use the same cost for processing the frame. The modules of dynamic face modeling and face recognition are implemented in child threads instead of the main thread. Thus, only the face tracking module is executed on every frame on the main thread.

2) Major Image Processing Modules:

- Image Preprocessing – Normalization, integral image calculationand gradient computation are applied on a frame.
- Face Detection – This module is conducted occasionally to adjust locations of tracked objects and to discover new objects in video as well. A fast, multi-view face detector is developed.
- Data Association – Detected faces in the frame are matched with existing objects in the target list. As the result, a new target list is generated with new or updated objects.
- Face/Head Tracking – For each object in the target list, its location is searched in the next frame. A particle filter based tracking approach is developed with adaptive models and constraints to prevent drift. Using OpenMP [12], multiple objects in the target list are searched in parallel in one frame.
- Dynamic Face Modeling – For people often appearing in recent video frames, a set of face models are generated automatically for each of these subjects along tracking which are saved in the memory. Then, when a subject is lost and later reappears, the query face will be matched with faces in the dynamic face collection first, then with faces in the static gallery. In this way, much faster response and more accurate result may be achieved for face identification. A mechanism is developed for this dynamic face modeling procedure.
- Face Recognition – This module is called for faces that are detected or tracked but not identified. A pretrained eye detector is first executed, and face recognition only applies when both eyes are localized with high confidence indicating the face is in a suitable pose for recognition. When running the module, a child thread is initiated for each face to be processed. That is, multiple child threads may be running at the same time on different faces in the same frame. Once a face is recognized, its identity is displayed in the subsequent frames along tracking until the target is lost.

IV. DEMO SOFTWARE EXAMPLE

The demo software, where the system supports manual tagging of an unknown subject. The user only needs to type the name of a subject appearing in the video once, then a face model of the subject is generated instantly and saved to the gallery. The subject may later have multiple face models in the dynamic collection saved in the memory to cover pose and lighting variations, which are created with proper face image samples automatically during tracking. Subjects and four subjects in the video respectively. It was also proved that with dynamic face modeling, a subject can be identified much faster when he/she reappears in the video. Some snapshots are presented in Fig.9 in which the blue bounding boxes indicatethe tracked locations of the subjects and the red tags show identities of the subjects derived from face recognition.

V. CONCLUSION

In this paper, the proposed framework is analyzed for acquiring super- resolution videos from low-resolution originals. Given that super-resolution conversion algorithms require a large amount of data processing, the proposed framework uses a set of strategies to improve performance and computational efficiency. The strategies consist of a combination of data simplification and parallel processing techniques. The simplification strategies are used to decrease the amount of data to process and, consequently, the required processing

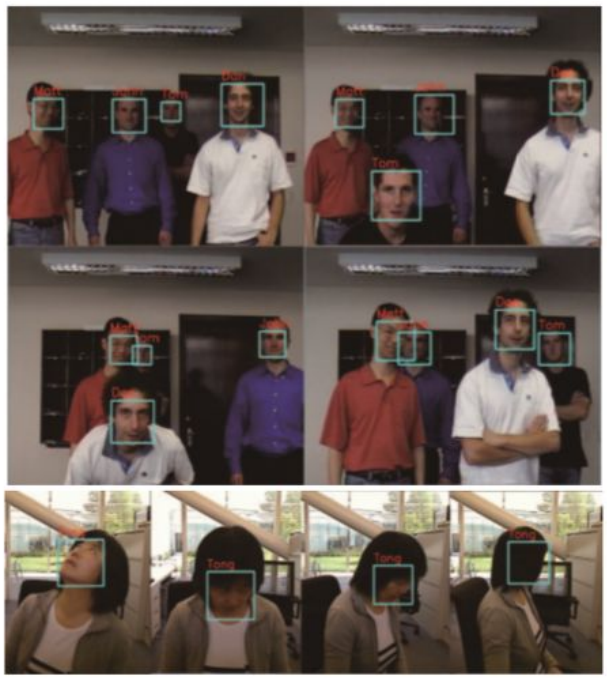

Fig. 9. Snapshots of running the demo software on video inputs.

time. The super-resolution videos are then used to detect objects in various ways by parallel processing. A demo software system was presented which allows instant recognition and tracking of multiple people in relatively from super-resolution videos. A parallel video processing pipeline was proposed with innovations made to image analysis modules for both efficiency and robustness.

ACKNOWLEDGMENT

The authors are grateful to Dr.S.Sridhar, Professor and Dean-Cognitive and Central Computing, Hemavathy.R, Assistant Professor CSE, of our college for the guidance to prepare this review article and also thankful to Dr.G.Shobha, HoD/CSE and Dr.N.K.Srinath, PG Dean (CSE) for the support offered.

VI. REFERENCES

[1] C. Weerasinghe, M. Nilsson, S. Lichman, and I. Kharitonenko, "Dig- ital zoom camera with image sharpening and suppression," Consumer Electronics, IEEE Transactions on, vol. 50, no. 3, pp. 777 – 786, aug. 2004.

[2] B. Lucas, T. Kanade, "An iterative image registration technique with an application to stereo vision," Intl' Joint Conf.on Artificial Intelligence,p.674-679, 1981

[3] D. Comaniciu, V. Ramesh, P. Meer, "Real-time tracking of non-rigid objects using mean shift," IEEE Conf. on CVPR, p.142-149, 2000.

[4] M. Arulampalam, S. Maskell, et al. "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," IEEE Trans. on Signal Processing, Vol.50, No.2, Feb. 2002.

[5] T. D. Ngo, D.-D. Le, et al."Robust face track finding in video using tracked points," IEEE Intl' Conf. on Signal Image Technology and Internet Based Systems, 2008.

[6] M. Kim, S. Kumar, V. Pavlovic, et al. "Face tracking and recognition with visual constraints in real-world videos," IEEE Conf.on CVPR, 2008.

[7] Z. Kalal, K. Mikolajczyk, et al. "Face-TLD: tracking-learning-detection applied to faces," IEEE Intl' Conf. on Image Processing, 2010.

[8] R. G. Cinbis, J. Verbeek, et al. "Unsupervised metric learning for face identification in TV video," IEEE Intl' Conf. on Computer Vision, p.1559–1566, 2011.

[9] S. Biswas, G. Aggarwal, P. Flynn, "Face recognition in low- resolution videos using learning-based likelihood measure- ment model," Intl' Joint Conf. on Biometrics, p.1–7, 2011.

[10] H. Bhaskar, "Integrated human target detection, identification and tracking for surveillance applications," IEEE Intl' Conf. on Intelligent Systems, p.467–475, 2012.

[11] Intel® TBB:https://www.threadingbuildingblocks.org/
OpenMP:http://openmp.org/wp/(Open Multi-Processing).