International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:6, No:11, 2012

Practical Issues for Real-Time Video Tracking

Vitaliy Tayanov

Abstract—In this paper we present the algorithm which allows us to have an object tracking close to real time in Full HD videos. The frame rate (FR) of a video stream is considered to be between 5 and 30 frames per second. The real time track building will be achieved if the algorithm can follow 5 or more frames per second. The principle idea is to use fast algorithms when doing preprocessing to obtain the key points and track them after. The procedure of matching points during assignment is hardly dependent on the number of points. Because of this we have to limit pointed number of points using the most informative of them.

Keywords—video tracking, real-time, Hungarian algorithm, Full HD video.

I. Introduction

Tracking objects on video is one of the important attributes of the complex problem of object/human behaviour analysis and interpretation. There exist a lot of algorithms of object/human action analysis which have been built mostly as hierarchical algorithms of analysis of actions starting from very simple ones to more complicated [1]. The most advanced of them use probabilistic approaches such as Graphical models (Bayesian Networks (BN), Dynamical Bayesian Networks (DBN) and Random Fields(RF) that have an important subclass of Markov Random Fields (MRF)). All such approaches do not use the information about object tracking and often utilise full information about object which is available on video frames. As known Graphical models usage could be very expensive in sense of computing when a number of random(hidden) variables is large [1], [3]. On the other hand often to compute a value containing in every random variable needs use all information from object and in case of Full HD video with high FR is critical.

Another approach that refers to behaviour analysis is based on analysis of tracks. In track building there exist several principle approaches. The most important are based on Kalman filter and particle filter as generalisation and extension to Kalman filter [2]. Both of them are of probabilistic character and can predict the dynamics of an object. The difference between them consists in limitations of Kalman filter that needs model be linear and noise needs be of Gauss character. Particle filter uses sampling of posterior distribution having observations and to track objects with its application in case of Full HD video and high FR is problematic.

One more group of tracking algorithms do not use any probabilistic on doing tracking. The algorithms from these group generate a number of keypoints every of them to be assigned to separate path if the assignment is valid. By the path we understand a sequence of points matched on two consecutive frames and all these points define a trajectory of an object. Among algorithms producing local feature points we can mention SURF (Speeded up Robust Feature), SIFT (Scale-Invariant Feature Transform) [2] and IPAN algorithms [6]. Because IPAN algorithm generally finds much less points and each point could be much more informative (in average) than point generated by SIFT and SURF we use it as feature generation algorithm for tracking.

IPAN algorithm finds keypoints on the contour of an object and belongs to geometrical approach to feature generating. First of all we separate background from image using Gaussian Mixture Model (GMM) [4] . This allows us to find zones of motion, i.e. objects that move and which will be used for further processing. To obtain contours of objects we do their segmentation before. This guaranties that we are going to have only external contours of objects and not internal ones that could be in case of Canny detector application [4], [5]. Canny filter uses gradients and finds all contours of an object. Keypoints are found in place where curvature of the contour segment is more than some given value. After finding critical points we use Hungarian algorithm for corresponding points assignment. This algorithm refers to the dynamic programming problem.

In the next sections we show how to use the algorithm which consists of different stages and some optimization tricks also. For any computing we use C++ for Windows (MS VS 2010) as well as for for Linux Ubuntu with libraries Open CV, Open GL, TBB, QT, BOOST, Open Threads and others.

II. Steps of Video Preprocessing

By video preprocessing we mean the following operations on video: background subtraction, object segmentation, contour detection, contour filtering and dominant points detection. For background subtraction we use three operations. First operation computes the foreground mask after we compute background image. Then we subtract this background from the grayscale image to obtain an image with moving objects on the black background.

After that we do operations on obtained image using functions from Open CV library to make segmentation of objects and then we find contours for segmented objects. For initially found contours we apply circular averaging filter to smooth contours. This is needed because of possible local sharpness of contours where the IPAN algorithm could find numerous of local dominants what are not informative at all. Such a filter realises the averaging operation both on two coordinates in a window of a given size. Then for filtered contours we apply IPAN algorithm. The function that realises such algorithm has 4 parameters. The principle idea of the algorithm consists in a describing the curvature by some keypoints named dominant points. Having such dominants we can connect them by lines of different orders thus having approximate contours. This

V. Tayanov, Ph.D. is Researcher in Polish-Japanese Institute of Information Technology (PJIIT), Bytom, 41-902, Poland e-mail:vtayanov@yahoo.com.

could be useful when we would like to realise the compression of information presented in contours. The total number of discrete points in contours is much more than the number of dominants, so the compression rate could be very high in some situations. It is very important to fix the parameters of IPAN algorithm in a way that produce dominants in the appropriate place. Normally they should be in place with high local curvature and the distance between points should be large enough to simplify matching. If the distance between any of dominants (in some feature space) is essentially more than the maximal shift of some point (for several consecutive frames) in the same feature space this should satisfy appropriate assignment. We consider the possible assignments during several consecutive frames because some dominants could disappear for some time due to contour changes.

It should be noticed that initial IPAN algorithm is not invariant to scale, i.e it is not invariant to the size of object in a video. To have dominants in appropriate place the sides of a triangle that should be placed in the internal segment of the contour depend on the size of this segment. Knowing this dependence we can make the re-scaling of the triangle. Fig.1 shows how to build the triangle inside the segment of a contour and all parameters used in the IPAN algorithm.

As we can see every triangle is characterised by coordinates of vertices p, $p+$ and $p-$. As coordinate features of a dominant point p we use (x_p, y_p) and two angles α and orientation angle B between side b and the mean line of the α angle of the triangle. Finally we have 4 features: two coordinates and two angles. The Euclidean distance between two dominants p_i from the previous frame and p_j from the next frame could be found as follows

$$d(p_i, p_j) = \Big(\sum_{k=1}^{N} (f_{ik} - f_{jk})^2 \Big)^{\frac{1}{2}} \tag{1}$$

where $f_i = \bigcup_{k=1}^{N} f_{ik}$ and $f_j = \bigcup_{k=1}^{N} f_{jk}$ are sets of features of p_i and p_j dominants. Here we have $N = 4$.

Having $d(p_i, p_j)$ we build so-called cost matrix that will be utilised for assignment using Hungarian algorithm. Actually p_i is the last point of each active track (see III).

However the predicting procedure could be improved by the following way. Let estimated position of the next point is $p_j^e = p_i + \vec{v_i}$, where $v_i = (v_i(x), v_i(y))$ is the previous velocity vector. So we can decompose p_j^e on $p_j^e(x) = p_i(x) + v_i(x)$ and $p_j^e(y) = p_i(y) + v_i(y)$. It should be noticed that if we have one point in some track than velocity of this point is equal to zero: $v_i = (0, 0)$. Then we use the following Euclidean distance to calculate the cost matrix:

$$\begin{aligned} d(p_j, p_j^e) = ((p_j(x) - p_j^e(x))^2 + (p_j(y) - p_j^e(y))^2 + \\ +(\alpha_i - \alpha_j)^2 + (B_i - B_j)^2)^{\frac{1}{2}} \end{aligned} \tag{2}$$

Finally we can predict the next speed value as

$$v_j = v_i + (p_j - p_i)\gamma \tag{3}$$

where $\gamma \in [0, 1]$. Prediction (3) could be written for separate coordinates x and y as well.

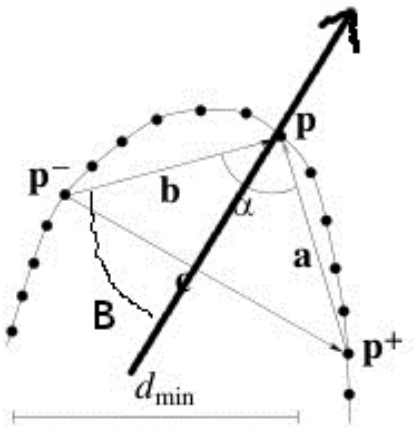

Fig. 1. Geometrical visualisation of IPAN algorithm

By doing recursion to compute the next value of the track velocity we have averaged velocity value of the track at each moment of time because of recursion. Such averaging works good if the person goes slightly with the approximately constant velocity. However if the person changes direction and the speed very often and fast this could lead us sometimes to significant errors. But in general situations this velocity prediction works sufficiently good and allows us to receive better tracks than standard way of cost matrix building.

III. Assignment and Tracking

A cost matrix has been built on the basis of two vectors of dominant points taken from the previous and the current frame. Dominants from the previous frame we call predicting points in sense of assignment of points on the current frame. This is because points from the previous frame have already been assigned and are the last points of each track (we assume that assignment has been taken place at previous stage). So if we have two sets of dominants s_{prev} (from the previous frame) and s_{next} (from the current frame) we can construct the cost matrix C of size $n \times m$, where $n = |s_{prev}|$ and $m = |s_{next}|$:

$$C_{i,j} = d(p_i, p_j), \tag{4}$$

where p_i is the ith point from the previous set of dominants and p_j is the jth point from the current (next relatively to previous frame) set of dominants. For finding assignments we use the Hungarian algorithm which works with cost matrix C. After application of this algorithm we obtain the binary matrix X composed with zeros and ones. Size of the cost matrix C should be $n \times n$ if $n \geq m$ or $m \times m$ if $m \geq n$. The realisation of Hungarian algorithm in C++ works in situation when $m \neq n$. This is done by adding columns or rows with zeros to achieve a cost matrix $\Big\{C_{ij}\Big\}_{n \times n}$ or $\Big\{C_{ij}\Big\}_{m \times m}$. A

binary matrix $\left\{X_{ij}\right\}$ of the same size that a cost matrix is filled as follows:

$$X_{ij} = \begin{cases} 1 & \text{for successful assignment;} \\ 0 & \text{otherwise.} \end{cases} \quad (5)$$

The properties of this binary matrix are as follows:

$$\begin{aligned} &\sum_{j=1}^{N} X_{ij} = 1 \forall i \in 1, ..., N; \\ &\sum_{i=1}^{N} X_{ij} = 1 \forall j \in 1, ..., N; \\ &\sum_{i=1}^{N} \sum_{j=1}^{N} C_{ij} X_{ij} \rightarrow \min \end{aligned} \quad (6)$$

where $N = \max(n, m)$.

The indexes i and j of each one in the binary matrix correspond to the dominants in the previous and the next sets that are candidates for assignment. The final decision about assignment will be made if the following condition is satisfied:

$$d(p_i, p_j) < d_{max}. \quad (7)$$

Distance d_{max} should be chosen as the maximal shift of an object (human) on the video with respect to FR. This could be done knowing maximal velocity of an object of interest with respect to the calibration parameters of video camera.

It should be noticed that algorithm that builds the paths should have the following functionality:

- the assignment should be executed only for active paths. By active path we understand some path which had no assignment not more than during several last frames (this parameter could be optimised during training process). Otherwise the path becomes inactive.
- The end of the track is the moment when the track is inactive and the last points of this track will not be checked for assignment.
- The points which have not been assigned are the beginnings of new tracks.

The optimised algorithm builds paths in a way that the length of such paths is as long as possible and there are not too many paths for each separate object.

IV. Some tests

We tested our algorithm on a video clip of 50 sec. with FR=25 taken from the video stream recorded from the Market Square of our town where a number of different activity actions can be registered. In Figs. 2-4 the main window with view of the Market Square from a single video camera is shown. All tracks have been built in these windows.

In Figs.5-6 tracking results from "Motion Capture" lab at PJIIT have been presented. Fig. 5 shows the initial scene and Fig.6 shows results of building of tracks. In Fig.7 we can see dominants found in the contours of two actors. As could be seen from Fig. 6 when person runs it is difficult to have "good" contours. This is because of limited frame rate. The simplest decision could be made by moving down the value of threshold of segmentation algorithm to have more clear

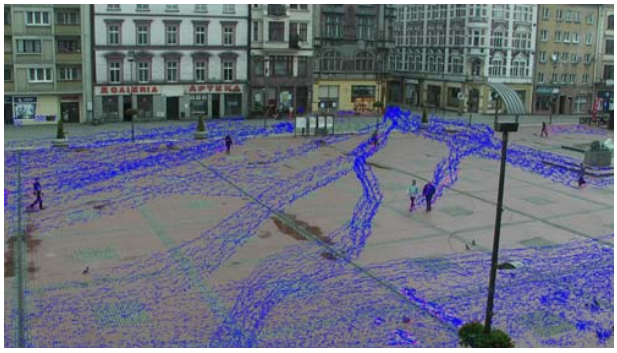

Fig. 2. Complete set of tracks plotted on the initial video of the Market Square

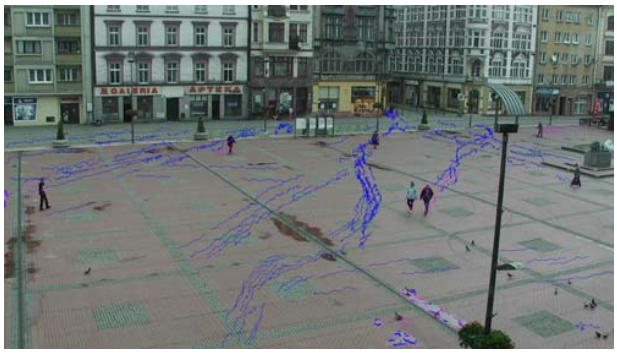

Fig. 3. Filtered set of tracks with the track length threshold equal to 25

contours but it can leads to appearance of shadows on the floor of the lab. Also this results in having "post-contours" when doing background separation. This is because of small movements of person or camera. This effect could be removed partially by setting the threshold to appropriate value.

In general case tracks and contours of every person should be rescaled on the basis of intrinsic and extrinsic parameters of a video camera. As seen from Figs. 2-4 the track building is sufficiently good. To that end we use filter on the track length. Such a filtering gives a possibility to see if the length of a given value is achieved for each person separately. As

Fig. 4. Filtered set of tracks with the track length threshold equal to 50

Fig. 5. Initial scene (”Motion Capture“ lab)

Fig. 6. Complete set of tracks (”Motion Capture“ lab)

seen even for the threshold of the track length equal to 50 we can see from one to several local tracks for some persons. For threshold equal to 25 we can see tracks for all persons. For threshold equal to 0 we can see all the tracks. The length of a track as one of the main characteristics of the track builder depends on the number of parameters. All of them we put on sliders (trackbars) to have the possibility to control them by a human. The sliders on the Control Panel (Fig.8) control the following parameters of the generalised algorithm:

- thresholds for the segmentation algorithm;
- size of a window for circular averaging filter;
- 4 parameters for the IPAN algorithm;

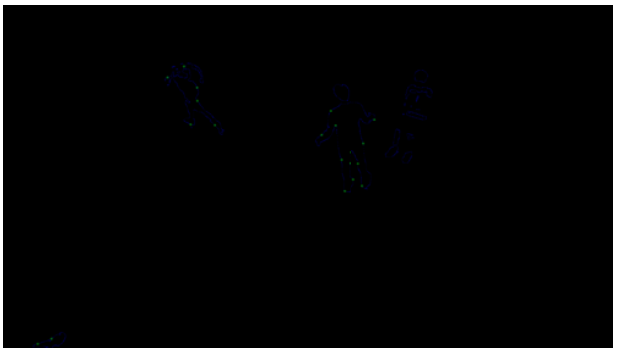

Fig. 7. Dominant points plotted on filtered contours (”Motion Capture“ lab)

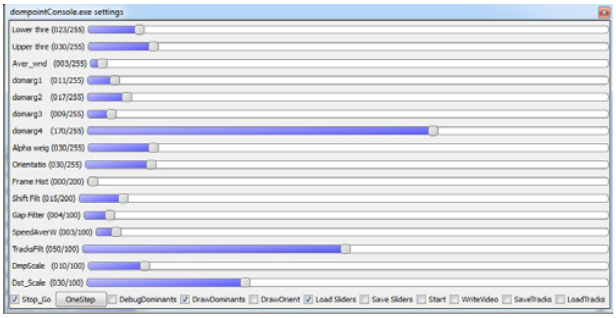

Fig. 8. Control panel

- weights on angles characterising dominants from the IPAN algorithm;
- d_{max};
- assignment gap (the number of frames where tracks are active if there is no assignment).
- window size to calculate average speed on the track end;
- filter for the track length;
- scale for the parameters of IPAN algorithm and d_{max} depending on the distance from the video camera.

V. Conclusion

We presented a generalised algorithm for video tracking that allows us to build tracks in a real-time or close to that for Full HD images. This is because it is based on geometrical approach for feature generating and fast enough assignment algorithm. A lot of things for acceleration of the algorithm could be done by changing of the parameters put on sliders. These parameters control the number of paths generated and also the length of such paths. Here improving the quality of paths (the number of paths reduction with obtaining paths of a larger length) we can speed up the algorithm.

Acknowledgement

This paper has been supported by the research project OR00002111: ”Application of video surveillance systems to person and behaviour identification and threat detection, using biometrics and inference of 3D human model from video.”

References

[1] Gong, Sh. and Xiang T., Visual Analysis of Behaviour: From Pixels to Semantics, Springer, London, 2011
[2] Maggio, E. and Cavallaro A., Video Tracking: Theory and Practice, Wiley, 2011
[3] Marsland, S., Machine Learning: An Algorithmic Perspective, ChapmanHall/CRC, Boca Raton, Florida, 2009.
[4] Prince, S., Computer Vision : Models, Learning and Inference, Cambridge University Press, 2012
[5] Bradski G. and Kaehler A., Learning OpenCV, O’Reilly, Sebastopol, CA, 2008
[6] Chetverikov, D. and Szabo, Zs., “A Simple and Efficient Algorithm for Detection of High Curvature Points in Planar Curves”, Robust Vision for Industrial Applications 1999, Vol. 128 ,1999, p 175–184.