Journal of Information Hiding and Multimedia Signal Processing ©2017 ISSN 2073-4212
Ubiquitous International Volume **8**, Number **5**, September **2017**

Tracking Multiple Moving Objects in Video Based on Multi-channel Adaptive Mixture Background Model

Dong-Ning Zhao[1,2]

[1]College of Information Engineering
Shenzhen University
Shenzhen, 518060, China
582101@qq.com

[2]Shenzhen Graduate School
Harbin Institute of Technology
Shenzhen, 518055, China
582101@qq.com

Da-Jie Guo[3], Zhe-Ming Lu[3*] and Hao Luo[3]

School of Aeronautics and Astronautics
Zhejiang University
Hangzhou, 310027, P. R. China
*Corresponding author: zheminglu@zju.edu.cn

Received July, 2016; revised April, 2017

Abstract. *This paper investigates tracking of multiple unknown objects in a video stream. Objects for tracking are added to the tracking list dynamically with the video going on. The task is to track all the moving objects in the video and obtain the trajectory of each object. In this paper, we propose a novel tracking strategy which can effectively locate a new moving object in the scene and its trajectory. Our strategy combines foreground extraction and optical flow points tracker. We extract the foreground using an improved Gaussian mixture model which can filter out the background. In order to determine the position of an object in latest frame, we calculate the displacement of objects' feature points in previous frame using pyramidal Lucas-Kanade tracker. The reliability of the tracker can be degraded by a variety of factors, including illumination change, poor image contrast, occlusion of feature. To track the object more accurately, we implement a forward-backward error filter to evaluate the quality of results. Besides the already existing objects, our method also can detect new intruding object and add it into the track list, we implement a foreground search procedure to realize it. We develop an improved mix background model which can extract the video stream foreground real-time for moving objects detection. At the end of this paper, we illustrate the efficacy of our method by showing the performance of an example tracking system with this strategy.*
Keywords: Optical flow, Tracking objects, Background model, Surveillance.

1. **Introduction.** A growing number of applications in surveillance systems and robotics and computer vision projects rely on real-time multiple object tracking. It is a great open challenge in computer vision to build a system that can kinematically track all objects in the scene. Many researchers have provided object tracking algorithms to develop a number of applications which include vehicle tracking[1], pedestrian monitoring[2], surveillance[3] and so on.

In general, there are three components in a tracking system: image representation, appearance model, and motion model, though in some cases these components are merged. In this paper we focus mainly in multiple moving objects detection and tracking since it is usually the most challenging to design.

Currently, the most popular method to track a target is to extract its features by using SSD and SIFT [4]. The next stage of all tracking schemes is to transform features to a feature vector and apply interest region detection to find the most matching region in current frame. Simple patch-matching schemes such as Sum of Squared Differences(SSD) do not perform well when subject to the errors introduced by interest region detectors. Other types of methods typified by SIFT[4] perform further processing on the extracted patches to compute a feature descriptor vector which is a scale invariant. Taylor [5] proposed a method based on simple pixel patches extracted from around interest points which were referred to as Histogrammed Intensity Patch(HIP). It is more robust in error registration because they employed a training phase to learn a model for the range of patches using independent per-pixel distributions.

Although many tracking methods employ appearance models that are either defined manually or trained by frames [6, 7, 8], these methods tend to have difficulties to handle high resolution video and multiple object tracking due to time-cost problems.

Distinguished from method above, Pyramidal LKT [9] is a quick and widely used algorithm which tracks the target by calculating the local multi-scale optical flow of the feature points to locate the target. Unfortunately, tracking often faces a problem where the points dramatically change appearance or disappear from the camera view in practice. Under such circumstances, tracking often results in failure. Kalal et al. [10] studied the problem of failure detection and proposed a forward-backward evaluation method. They implemented a tracker to produce a trajectory by tracking the points forward in time and a trajectory reversely, then compared two trajectories to filter the significantly different points. They also proposed a novel tracking framework TLD [11]. TLD contained three subdivisional tasks: tracking, learning, and detection. Their method achieved good performances in tracking the single object which was initially located in first frame, but it is not well competent to handle the multiple objects tracking tasks.

Previously, Guo et al. proposed an improved real-time mix background subtraction model[12]. In this paper, we will propose a novel multiple objects tracking framework based on foreground extraction. The framework decomposed the multiple objects tracking into three subtasks: foreground extraction, tracking, and detecting the new object intruding in the scene. The foreground extraction procedure locates the interested regions. The tracking procedure follows objects frame by frame. The detecting procedure finds new coming objects to track.

The remainder of the paper is organized as follows: Section 2 reviews the previous work related object tracking methods. Section 3 introduces our multiple objects tracking framework. Section 4 performs some test results and draws a conclusion. Section 5 analyzes the merits and demerits of our framework, and then finished with the plan of the future research.

2. **Related Work.**

This section reviews the related part of our system. Section 2.1 reviews the improved mix background model for real-time tracking we previously proposed. Section 2.2 reviews the Pyramidal Lucas Kanade Tracker. Section 2.3 reviews the Forward-Backward error detector.

2.1. **Multi-channel Adaptive Mixture Background Model.**

Extracting the foreground regions in image sequences is the fundamental task of our tracking framework.

Stauffer and Grimson[13] employed an adaptive nonparametric Gaussian mixture model to solve these problems. Their model can suppress the effect of small repetitive motions such as moving branch and leaves of trees as well as small camera movement. Zivkovic [14] proposed an improve method to cancel the influence of the prior video frames, and he invented an auto changed segment number algorithm to adapt to each pixel, so the processing time of each frame is reduced.

To improve the performance, we develop a multi-channel adaptive background model to extract the interested areas. We model every pixel in a frame by a mixture of maximum K Gaussian segments.

Definition 2.1.1 At any time t, the sequence of a pixel at (x, y), which represents the position at the y-th row and x-th column, can be expressed like that:

$$\{X_1, X_2, ..., X_t\} = \{I_i(x, y) : 1 \leq i \leq t\} \tag{1}$$

Definition 2.1.2 The probability of this pixel value in time t appearing is:

$$P(X_t) = \sum_{i=1}^{K} w_i \eta(X_t, \mu_i, \sigma_i^2) \tag{2}$$

Where w_i is the estimate weight parameter of the i-th Gaussian segment. $\eta(X_t, \mu_i, \sigma_i^2)$ is the normal distribution probability density function of the i-th segment.

Definition 2.1.3 The normal distribution probability density function is shown below:

$$\eta(X, \mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} \exp(\frac{-(x-\mu)^2}{2\sigma^2}) \tag{3}$$

Remark 2.1.1 It should be noted that all distributions are ordered by the parameter called fitness value,

$$fitness = \frac{w}{\sqrt{\sum_{i=1}^{N} \sigma_i^2}} \tag{4}$$

the first B segments are chosen as the background model, where B is defined as follow,

$$B = \arg\min_b (\sum_{k=1}^{b} w_k > T) \tag{5}$$

At one time, only the matched one is updated, and other segments maintain the last state.

Definition 2.1.4 The multi-channel update equations are shown below:

$$\hat{w}_{i,t+1} = \hat{w}_{i,t} + \alpha(1 - \hat{w}_{i,t}) \tag{6}$$

$$\hat{\mu}_{i,t+1} = \hat{\mu}_{i,t} + \alpha(x_{t+1} - \hat{\mu}_{i,t}) \tag{7}$$

$$\hat{\sigma}_{i,t+1}^2 = \hat{\sigma}_{i,t}^2 + \alpha((x_{t+1} - \hat{\mu}_{i,t})^2 - \hat{\sigma}_{i,t}^2) \tag{8}$$

Example 2.1.1 Let us consider the following example, if one pixel contain R, G, B three channels, $X = (x_1, x_2, x_3)$, our method will record and update respective mean and variance using Eqs. (7) (8), but only one weight for a pixel using Eq. (6).

Definition 2.1.5 In order to check the matching result, we define a matching judging condition:

$$\sum_{c=1}^{N} (x_{t+1,c} - \hat{\mu}_{t,c})^2 < N(2.5Var_T)^2 \tag{9}$$

where c is the channel number of video.

When the update procedure is completed, the weight must not sum to one, so our algorithm implements a normalization. If no one is matched, a new distribution is built, if the number of segment reaches the limit, the least weight component will be replaced.

2.2. **Pyramidal Lucas-Kanade Tracker.** Since the Lucas-Kanade algorithm [15] was proposed in 1981 image alignment has become one of the most widely used techniques in computer vision. Bouguet [9] proposed the pyramidal implementation of Lucas-Kanade algorithm. It can calculate the optical flow quickly and accurately and become a most widely used tracking method. In the following part, we will review the pyramidal Lucal-Kanade algorithm briefly.

Definition 2.2.1 Let I and J denote two gray images adjacent to each other in a video sequence. Then $I(\boldsymbol{x}) = I(x, y)$ and $J(\boldsymbol{x}) = J(x, y)$ denote the pixel value of two images at the location $\boldsymbol{x} = (x, y)^T$, where x and y are column and row position number. The image I is referenced as the previous frame, and the image J as the current frame. Generally, the top left corner pixel coordinate vector is $(0, 0)^T$ (origin point). Let W and H denote the width and height of two images, so the bottom right point position is $(W-1, H-1)^T$.

Consider an image pixel $\boldsymbol{a} = (a_x, a_y)^T$ on the previous image I, the task of this algorithm is to find the most possible location $\boldsymbol{b} = (b_x, b_y)^T = (a_x + d_x, a_y + d_y)^T$ on the current frame J. The vector $\boldsymbol{d} = (d_x, d_y)^T$ represents the velocity of point at an optical flow. In addition to a displacement component $\boldsymbol{d}$, the algorithm assumes that the image undergoes an affine deformation between two frames in the neighborhood of the two points $\boldsymbol{a}$ and $\boldsymbol{b}$.

Corollary 2.2.1 Under the assumption above, it is natural to introduce an affine transformation matrix $\mathbf{A}$:

$$\mathbf{A} = \begin{pmatrix} 1 + d_{xx} & d_{xy} \\ d_{yx} & d_{yy} \end{pmatrix} \tag{10}$$

Where the four coefficients d_{xx}, d_{xy}, d_{yx} and d_{yy} describe the affine deformation of the image patch. The task is transformed to find the displacement vector $\boldsymbol{d}$ and affine matrix $\mathbf{A}$.

Definition 2.2.2 To solve this problem, we should minimize the residual function $\varepsilon(\boldsymbol{d}, \mathbf{A})$ defined as follow:

$$\varepsilon(\boldsymbol{d}, \mathbf{A}) = \varepsilon(d_x, d_y, d_{xx}, d_{xy}, d_{yx}, d_{yy}) = \sum_{x=-w_x}^{w_x} \sum_{y=-w_y}^{w_y} (I(\boldsymbol{a} + \boldsymbol{x}) - J(\boldsymbol{a} + \mathbf{A}\boldsymbol{x} + \boldsymbol{d}))^2 \tag{11}$$

where w_x and w_y define the size of the local image window and $\boldsymbol{x} = (x, y)^T$. The typical value of w_x and w_y are $3, 5, 7, 10$. Let L denote the pyramidal image level. For $L = 0, 1, 2, \ldots, L_m$, and generally L_m is set to 4 or 5 depending on the resolution of frame. Then build the image pyramidal by down sample the image I and J by the rate 2. Following the definition above, the vector $\boldsymbol{a}_L = \boldsymbol{a}/2^L$ and similarly for $\boldsymbol{b}_L, W_L$ and H_L.

The overall pyramidal tracking algorithm proceeds as follows: firstly, the optical flow and affine transformation matrix are computed at the deepest pyramid level L_m. Then, the result of that level is propagated to the upper level L_{m-1} as the initial assumption for the displacement vector and affine matrix and repeat the computation process to find the result of this level. Repeat the propagation and computation procedure until up to level 0 which represents the origin scale of the frame. For expression convenience, we use $I(\boldsymbol{x})$ to denote a pixel around point $\boldsymbol{a}$, similarly for $J(\mathbf{A}\boldsymbol{x} + \boldsymbol{d})$. So the error function is transform like that:

$$\varepsilon(\boldsymbol{d}, \mathbf{A}) = \sum_{x=-w_x}^{w_x} \sum_{y=-w_y}^{w_y} (I(\boldsymbol{x}) - J(\mathbf{A}\boldsymbol{x} + \boldsymbol{d}))^2 \tag{12}$$

Proof: Let D be the derivative matrix of the error loss function above,

$$D = (\frac{\partial \varepsilon}{\partial d_x}, \frac{\partial \varepsilon}{\partial d_y}, \frac{\partial \varepsilon}{\partial d_{xx}}, \frac{\partial \varepsilon}{\partial d_{xy}}, \frac{\partial \varepsilon}{\partial d_{yx}}, \frac{\partial \varepsilon}{\partial d_{yy}}) \tag{13}$$

When D reaches the state below, we get optimized parameters.

$$D_{optimum} = (0, 0, 0, 0, 0, 0) \tag{14}$$

We expand the expression of D:

$$D = -2 \sum_{x=-w_x}^{w_x} \sum_{y=-w_y}^{w_y} (I(\boldsymbol{x}) - J(\mathbf{A}\boldsymbol{x} + \boldsymbol{d}))P(J, \boldsymbol{x}) \tag{15}$$

Where $P(J,\boldsymbol{x})$ is the differential matrix of $J(\mathbf{A}\boldsymbol{x} + \boldsymbol{d})$:

$$P(J, \boldsymbol{x}) = (\frac{\partial J}{\partial x}, \frac{\partial J}{\partial y}, x\frac{\partial J}{\partial x}, y\frac{\partial J}{\partial x}, x\frac{\partial J}{\partial y}, y\frac{\partial J}{\partial y}) \tag{16}$$

According to Taylor expansion theory, we obtain an approximate substitution:

$$J(\mathbf{A}\boldsymbol{x} + \boldsymbol{d}) \simeq J(\boldsymbol{x}) + P(J, \boldsymbol{x})\boldsymbol{d} \tag{17}$$

Although $P(J,\boldsymbol{x})$ is defined as the differential matrix computed from image J, we can also perform that computation from image I since the motion between the two images is expected to be small. As a result, we perform another approximate substitution:

$$D \approx -2 \sum_{x=-w_x}^{w_x} \sum_{y=-w_y}^{w_y} (I(\boldsymbol{x}) - J(\boldsymbol{x}) - P(I, \boldsymbol{x}))P(I, \boldsymbol{x}) \tag{18}$$

The remain part is a mathematical problem, you should simply do the expression first and use Newton-Raphson iterative algorithm to obtain the accurate approximation.

2.3. **Forward-Backward Error Detector.** As we can see from Section 2.2, the pyramidal LK optical flow tracker performs without any failure detection, although tracking often results in failure in practice. Kalal et al [10] proposed a method that enabled any tracker to self-evaluate its reliability. The proposed method is based on so called forward-backward consistency assumption that correct tracking should be independent of the direction of time-flow. The overall forward-backward error detector performs as follows. Firstly, the tracker produces the points' location in the current frame by tracking forward in time. Secondly, the location of the validation points in the previous frame is calculated by backward tracking from the current frame to the previous frame. Thirdly, compare the original location and the validation location, if they differ significantly, the forward tracking is regarded as failure.

Definition 2.3.1 Let $S = (I_1, I_2, \ldots, I_t, I_{t+1}, \ldots, I_{t+k})$ be an image sequence and x_t be a point location in time t. Using LKT to obtain the forward tracking location $\hat{x}_{t+1}$ on I_{t+1}, then track the point $\hat{x}_{t+1}$ from I_{t+1} to I_t to produce the validation location $\hat{x}'_t$. The forward-backward error is defined as

$$FBE = \|\hat{x}'_t - x_t\| \tag{19}$$

3. **Multiple Objects Tracking Framework.** Multiple objects tracking framework is designed for tracking multiple unknown objects in a video. The block diagram of the framework is shown in Fig. 1.

This framework consists of four parts. The foreground extraction part detects the moving objects in the scene for the next procedure. The tracking list records all the objects' locations for tracking and the trajectories. The LK tracker estimates the objects' motion between consecutive frames under the assumption that the frame-to-frame motion is limited and the objects are visible. The forward-backward error maps are generated by LK tracker with a brute force points selection mechanism, which selects feature points using uniform under-sample. When a new frame comes, it is processed by our tracking

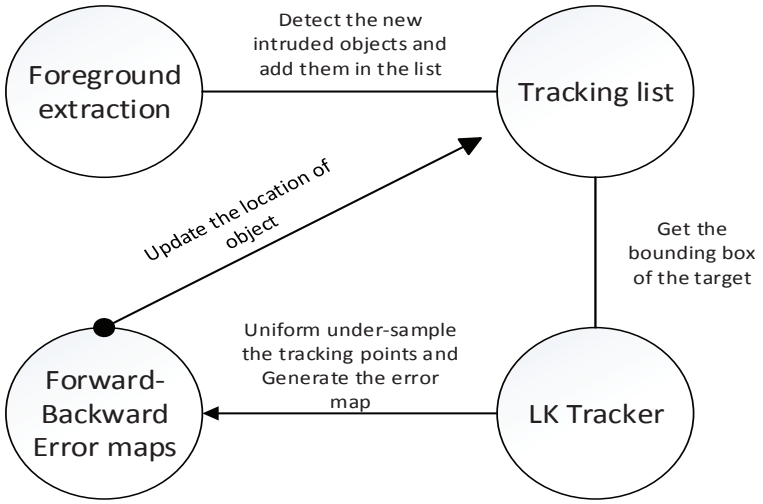

FIGURE 1. The General framework for multiple objects tracking.

system by using foreground extractor and then the interested regions in this frame are located. Then we estimate and update the locations of objects already in tracking list, and compare the foreground regions and the regions in the tracking list. If some foreground regions are not in the tracking list, we add them to the list as new targets. When an existing target doesn't match with any foreground region, we add a time stamp to this target. If it stays static over a threshold time, the system will delete it from the tracking list. If some targets meet tracking failure, we also delete them.

To estimate the bounding boxes of targets, we implement a median filter. When we get a forward-backward error map composed by the FBE of sample points, we search the median in this map and filter out the larger half. Then we calculate the average displacements of the remaining point as the bounding box displacement. To calculate the scale change, we calculate the current and previous distances for each remaining pair and obtain a ratio between them, and then the median of all ratios is defined as the bounding box scale change ratio. In comparing procedure, an important question is how to define two regions as matching. We use overlap ratio to solve this problem.

Definition 3.1 If A_1 is the area of a foreground region, A_2 is the area of an existing target region and A_{12} is the overlap area of two regions, we define overlap ratio as:

$$OR = \frac{A_{12}}{A_1 + A_2 - A_{12}} \tag{20}$$

If $OR > 0.3$, we consider the two regions are matching.

4. **Experimental Results.** In this Section, we do some experiments to analyze the performance of our multiple objects tracking framework and draw a conclusion about the results. We select three real indoor surveillance video fragments for experiment, which contain one scene of a corridor and two scenes of a gate. The experimental program runs at a 2.93GHz intel E7500 PC. Fig. 2 shows that two people go into the scene, the right one walks into the room and the left one goes straight. In Fig. 3, two pedestrians walk side by side from left to the gate. In Fig. 4, three pedestrians come in the hall from the gate, then two walk to the top left and another walk to the bottom right. To distinguish the different objects, our tracking system gives each object an ID tag which is marked at the top left of the object, and we also display the trajectory of each target using a red line. As you can see in Figs. 2-3, our system work well at three surveillance scenes.

FIGURE 2. Corridor.

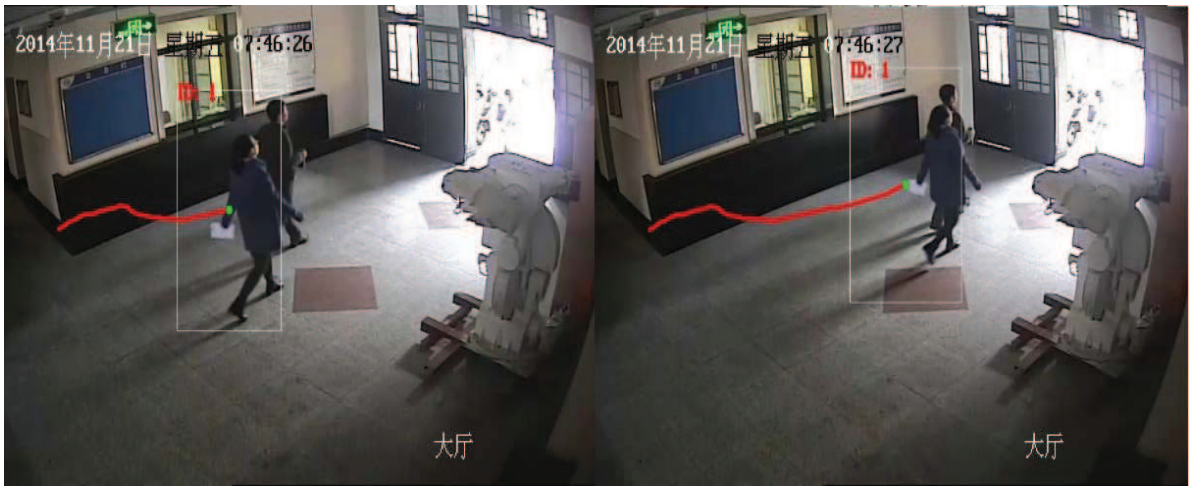

FIGURE 3. Gate 1.

Because the resolution of the selected video sequences is relatively high (960×576), the time cost is a little higher, achieving only quasi real-time level. Table 1 shows the average processing time cost per frame. When the number of objects increases, the time consumption also increases.

To evaluate the performance of the video tracking, manually moving objects are extracted as the ground truth and compared with the results obtained by the proposed method. From each frame in the video, few parameters like True Negative (TN), True Positive (TP), False Negative (FN) and False Positive (FP) are calculated. Where TP is total number of pixels where both ground truth and proposed results overlap, TN is total number of pixels that both ground truth and proposed results doesn't contain object, FP is total number of pixels, where obtained results contain object, while ground truth does not contain any object and FN is total number of pixels where ground truth contains the object and proposed system does not contain the object. Later, the metrics like Recall and Precision are calculated using TP, TN, FP and FN. The Recall also known as detection rate gives the percentage of matching true positive positives with ground truth

FIGURE 4. Gate 2.

TABLE 1. Time consuming results.

Sequence	Corridor	Gate 1	Gate 2
Average time cost(ms)	123.6	140.2	150.5

TABLE 2. Recall and precision results.

Sequence	Corridor	Gate 1	Gate 2
Average recall	0.7652	0.7932	0.8143
Average precision	0.8331	0.8523	0.8612

as Recall = TP/(TP + FN). Precision represents the percentage of true positives not matching with ground truth as Precision = TP/(TP + FP). Table 2 shows the average tracking recall and precision per frame. For all video sequences, our scheme can get the tracking recall and precision around $76\% - 86\%$.

5. **Conclusions.** In this paper, we studied the problem of tracking multiple objects in a video stream. We designed a new framework that tacking multiple objects based on foreground extraction. A quasi real-time implementation of this strategy has been described in detail. And we performed some experiments to illustrate the efficacy of our framework. Though we achieve a fundamental efficacy, there are a number of limitations in this framework. For instance, our strategy is based on foreground extraction. If the foreground is unreliable, the tracking system will fall into chaos. And the system only performs well with videos recorded by static cameras. Another problem is that our framework doesn't contain any learning components. If an object disappears in the scene and returns after a period, our system will not recognize it and will treat it as a new object. And if two or more objects are too close to be separated, as shown in Fig. 3, our system will regard them as one object. It will cause some system unreliability. In the future, we will study some learning methods to improve the performance, and it will a big challenge for time consumption and accuracy.

Acknowledgments. This work was supported partially by the Nature Science Foundation of Guangdong Province (2015A030310172), the Science & Technology Plan Projects

of Shenzhen (JCYJ20150324140036830, GJHZ20160226202520268).This work was also partly supported by the Zhejiang Provincial Natural Science Foundation of China (Grant No. LY17F030008, No. LY15F010003 and No. LY14F020024), the Public Good Research Project of Science and Technology Program of Zhejiang Province, China. (NO.2016C31097) and the Science and Technology Project of QuZhou, Zhejiang, China.(NO.2015Y005).

REFERENCES

[1] D. Koller, J. Weber, and J. Malik, Robust multiple car tracking with occlusion reasoning. *Lecture Notes in Computer Science*, vol 800, Springer, Berlin, Heidelberg, pp. 189–196, 1994.

[2] R. Rosales, and S. Sclaroff, Improved tracking of multiple humans with trajectory prediction and occlusion modelling, *Proceedings of the IEEE Workshop on the Interpretation of Visual Motion*, pp. 1–8, 1998.

[3] A. M. Baumberg, D.C. Hogg, An efficient method for contour tracking using active shape models, *Proceedings of the IEEE Workshop on Motion of Non-Rigid and Articulate Objects*, pp. 194–199, 1994.

[4] D. G. Lowe, Distinctive image features from scale-invariant key points, *International Journal of Computer Vision*, vol.60, no. 2, pp. 91–110, 2004.

[5] S. Taylor, and T. Drummond, Multiple Target Localisation at over 100 FPS, *Proceedings of the British Machine Vision Conference*, pp.58.1-58.11, 2009.

[6] P. Tissainayagam, and D. Suter, Object tracking in image sequences using point features, *Pattern Recognition*, vol. 38, no. 1, pp.105–113, 2005.

[7] P. Tissainayagam, and D. Suter, Assessing the performance of corner detectors for point feature tracking applications, *Image and Vision Computing*, vol. 22, no. 8, pp. 663–679, 2004.

[8] S. Stalder, H. Grabner, and L. V. Gool, Beyond semi-supervised tracking: Tracking should be as simple as detection, but not simpler than recognition, *IEEE 12th International Conference on Computer Vision Workshops*, pp. 1409–1416, 2009.

[9] J. Y. Bouguet, Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm, *Intel Corporation*, vol. 1, no. 2, pp. 1–9, 2001.

[10] Z. Kalal, K. Mikolajczyk, and J. Matas, Forward-backward error: Automatic detection of tracking failures, *IEEE 20th International Conference on Pattern Recognition*, pp. 2756–2759, 2010.

[11] D. J. Guo, H. Luo, and Z. M. Lu, Multi-Channel Adaptive Mixture Background Model for Real-time Tracking,*Journal of Information Hiding and Multiple Signal Processing*, vol. 7, no.1, pp. 216–221, 2015.

[12] Z. Kalal, K. Mikolajczyk, and J. Matas, Tracking-learning-detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no.7, pp. 1409–1422, 2012.

[13] C. Stauffer, and W. E. L. Grimson, Adaptive background mixture models for real-time tracking, *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 2, pp. 252, 1999.

[14] Z. Zivkovic, Improved adaptive Gaussian mixture model for background subtraction, *Proceedings of the 17th International Conference on Pattern Recognition*, Vol. 2, pp. 28–31, 2004.

[15] B. D. Lucas, and T. Kanade, An iterative image registration technique with an application to stereo vision, *International Joint Conference on Artificial Intelligence*, vol. 81, pp.674–679, 1981.