

Multi-Person Recognition Using Viola-Jones and Kalman Filter for Target Tracking

Sreelekha M¹, Sufyan P²

¹PG Student, Computer Science and Engineering, MEA Engineering College, Kerala, India

²Assistant Professor, Computer Science and Engineering, MEA Engineering College, Kerala, India

Abstract: Multi-person tracking is a summary of single trackers. Everyone's motion is independent of others. This paper presents a multi-level framework for target tracking in simple and complex environments. Kalman filter is used to perform target tracking. The Kalman filter tracks people even if their spots merge, provide more efficiency to track multiple people. It provides the best estimate of its position in each time step. If all the noise is Gaussian, then the optimality is guaranteed. Based on position estimates, KF gives better results to avoid occlusion. For each object in the scene, a separate KF is initialized and its trajectory is modeled. The first phase detects all the faces in the crowded scenes and second phase detects the face of the target person using Viola-Jones algorithm for face recognition and third phase tracks the target person from that crowd.

Keywords: Kalman Filter, Occlusion, Target Tracking, Viola- Jones Algorithm

1. Introduction

Visual tracking has been an important issue in computer vision, video surveillance, vehicle navigation, motion analysis and human-computer interface applications. Although a number of tracking methods have been proposed, they are still a challenging problem due to factors such as partial occlusion, illumination changes, posture changes, background clutter and viewpoint change. As shown in Fig. 1, most visual tracking methods comprise object model, information integration, target detection and model update. For different methods, different processes are emphasized.

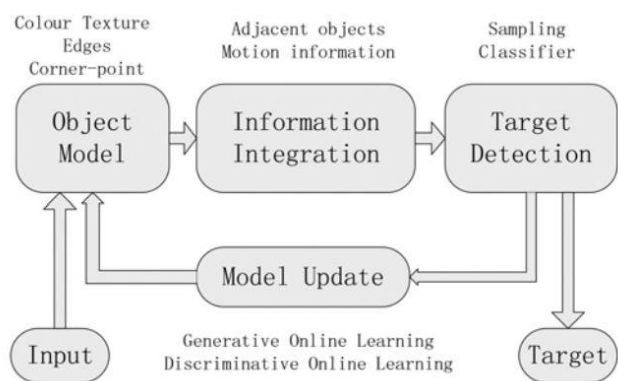


Fig -1: Flowchart of Visual Tracking

Detecting human faces in a video is also a great challenging problem. These configurations may be like angle of view, background intensity, and various illuminations. This is due to high variety of configurations that may occur. The complexness of the face results in a particular degree of issue for fast detection and tracking [1].

Tracking is the process of locating moving objects in time. An algorithm analyzes each frame and outputs the location of moving targets within the video frame. This algorithm has two basic steps - detection of moving objects (here people) in each frame and tracking them in each consecutive frame.

Detection of objects is a bottom up approach and has low computational complexity. Tracking is basically a top down process and have higher computation complexity. Here Kalman filter algorithm is used for tracking moving objects and Viola Jones algorithm is used for detection of face of the target person.

The purpose of the research is to develop real-time computer vision algorithms in order to track targets in uncertain complex environments. Two major subjects of this work are:

- The development of fast and robust image processing algorithm that able to track target person.
- The development of an algorithm which automatically detect aerial target in a given video.

2. Literature Survey

Comparing different target tracking and face recognition methods used for visual tracking.

2.1 Comparison of Target tracking algorithms

Tracking is the process of object of interest within a sequence of frames, from its first appearance to its last. The type of object and its description within the system depends on the application. During the time that it is present in the scene it may be occluded by other objects of interest or fixed obstacles within the scene. A tracking system should be able to predict the position of any occluded objects. Object tracking systems are typically geared towards surveillance application where it is desired to monitor people or vehicles moving about an area.

2.1.1 Particle Swarm Optimization

The PSO algorithm was first introduced in [2]. The term swarm is applied to fish, insects, birds, and microorganisms, and describes a behavior of a school of animals of similar size and body orientation, generally cruising in the same direction. These animals adjust their physical movement in

order to avoid predators, seek food and mates and optimize environmental parameters such as temperature.

The general idea of PSO is inspired by a flying swarm of birds searching for food: they will cooperatively search the area by continuously communicating their current positions and proximity to one another. Every bird knows where the other birds are and what they have found. The reliability of a recently found food location is defined by a static evaluation function, the so called fitness function. In every round of the algorithm, birds cogitate whether it is appropriate to change their position towards the higher amount of food or to stay at their current position. After much iteration, nearly all birds are at the peaks or close to the peaks of the search space where the food exists.

In comparison with other stochastic optimization techniques like genetic algorithms (GAs) [3], PSO have fewer complicated operations, fewer defining parameters, and can be coded in just a few lines. Because of these advantages, it has received increasing attention in a variety of applications in recent years. Training of artificial neural networks, finding optimal solutions for equations, and localizing mobile robots in video sequences [4] [5] are few examples. Basically, PSO is usable for every problem definable with an n-dimensional search space, where every position in this search space has a testable value. In PSO, the particles are initialized with certain positions of the objects to be tracked. The influence the particles exercise on each other is based on changing factors like the position of those particles, the direction they are flying to and their velocity. The velocity of each particle is defined by

$$v_{x,y,z}(t) = w \cdot v_{x,y,z}(t-1) + c_1 \cdot \phi_1 \cdot (pbest_{x,y,z} - P_{x,y,z}(t-1)) + c_2 \cdot \phi_2 \cdot (gbest - P_{x,y,z}(t-1))$$

where $v_{x,y,z}(t)$ denotes the velocity in the x, y, and z directions at time instant t, w is a weighting factor, c_1 and c_2 are a-priori defined weighting factors, ϕ_1, ϕ_2 are random numbers between 0 and 1, p is the position of the particle, pbest is the best position found by this particle and gbest is the best position found by all particles.

The position at the current time t for a certain particle $P_{x,y,z}(t)$ is updated as its previous position $P_{x,y,z}(t-1)$ augmented by its current velocity $v_{x,y,z}(t)$. This process is repeated after every iteration of the PSO algorithm.

$$P_{x,y,z}(t) = P_{x,y,z}(t-1) + v_{x,y,z}(t)$$

It is important that the search space shows convergent behavior; otherwise it is nearly impossible for the PSO to attain acceptable results. The PSO mechanism assumes that in most situations the peaks have smaller peaks around them. This concept is inspired by nature: If the highest mountain top is at a certain location, it is normally surrounded by lower mountain tops around that location. Thus, if one particle has found the current global best position, all the other particles drift towards this best particle. The velocity and the accurate direction of a drifting particle changes from one frame to the other, it's therefore not deterministic. Fig. 2 illustrates a

typical search scenario for the global best position in a three-dimensional space.

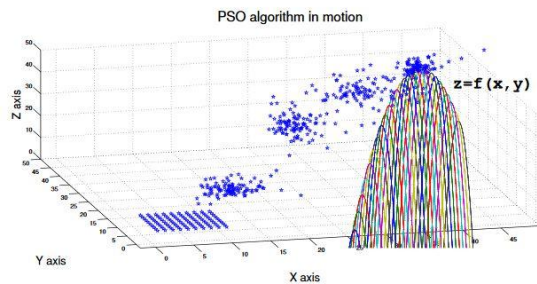


Figure 2: Different time-snaps of one swarm of particles searching for the global best position in a three dimensional space

The term $z = f(x,y)$ denotes a randomly generated function with peaks distributed according to random variables x and y. In this context, (x,y) could pinpoint the position of the particle with respect to the object to be tracked, and $f(x,y)$ denotes the grey value of the object at this position. The best position (x,y) corresponds to the highest point in $f(x,y)$. All the N particles assigned with a certain object to be tracked are communicating their current position and trying to find the global best position.

2.1.2 Sparse Representation and L1 minimization

In paper [6], the author develops a robust visual tracking framework by transforming the tracking problem into finding a sparse approximation in the template subspace, and proposes using trivial templates to deal with the occlusion, so that each trivial template has only one non-zero element. Then, during the trace, the target candidate is represented as a linear combination of template templates consisting of two target templates (obtained from the previous frame) and trivial templates. The number of target templates is far less than the number of ordinary templates. Intuitively, a good target candidate can be effectively represented by the target template. Since coefficients corresponding to trivial coefficients tend to be zero, this results in a sparse coefficient vector. In the case of occlusion a limited number of trivial coefficients will be activated, but the entire coefficient vector will remain sparse. In contrast, bad target candidates often result in dense representations. The sparse representation is achieved by solving the l_1 regularized least squares problem, which can be done efficiently by convex optimization. The candidate with the minimum target template projection error is then selected as the tracking result. The tracking is then guided by the Bayesian State Inference Framework, where the particle filter is used to propagate the sample distribution over time. This approach includes two additional components to further enhance robustness. First, they perform a non-negative constraint on the sparse representation. These constraints are particularly useful for eliminating clutter similar to a target template having a reverse intensity pattern. Constraints are implemented by including positive and negative generic templates in the template set. Second, the target template set is dynamically updated to maintain a representative template throughout the tracking process. This is done by using the coefficients in the sparse representation to adjust the template weights.

2.2 Comparison of Face Recognition algorithms

In the age of rising crime face recognition is enormously important in the contexts of computer vision, psychology, surveillance, fraud detection, pattern recognition, neural network, content based video processing, etc. Face is a non intrusive strong biometrics for identification and hence criminals always try to hide their facial organs by different artificial means such as plastic surgery, disguise and dummy.

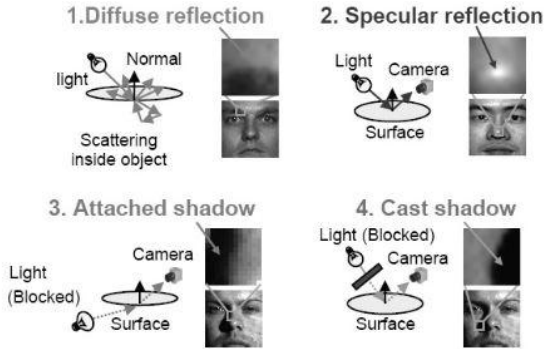


Figure 3: Variation of facial appearances in case of Face Recognition

Fig. 3 shows the variation of facial appearances in case of face recognition. The availability of a comprehensive face database is crucial to test the performance of these face recognition algorithms.

2.2.1 Principal Component Analysis

PCA commonly uses the eigenfaces [7] [8] in which the probe and gallery images must be the same size as well as normalized to line up the eyes and mouth of the subjects whining the images. Approach is then used to reduce the dimension of data by the means of image compression basics [9] and provides most effective low dimensional structure of facial pattern. This reduction drops the un useful information and decomposes the face structure into orthogonal (uncorrelated) components known as eigenfaces. Each face image is represented as weighted sum feature vector of eigenfaces which are stored in 1-D array. A probe image is compared against the gallery image by measuring the distance between their respective feature vectors then matching result has been disclosed. The main advantage of this technique is that it can reduce the data needed to identify the individual to $\frac{1}{1000}$ of the data presented [10].

The basis vector are computed from the set of training images I . The average image in I is computed and subtracted from the training images, creating set of data samples

$$i_1, i_2, \dots, \dots, i_n \in I - \bar{I}$$

These data samples are arrange in a matrix represented as

$$x = \left(\begin{pmatrix} \vdots \\ i_1 \\ \vdots \end{pmatrix} \dots \begin{pmatrix} \vdots \\ i_n \\ \vdots \end{pmatrix} \right)$$

XX^T is then the sample covariance matrix for the training images and the principal components of the covariance matrix are computed by solving $R^T XXX^T R = \Lambda$ where Λ the diagonal matrix of eigenvalues is and R is the matrix of orthonormal eigenvectors. Geometrically, R is a rotation matrix that rotates the original coordinate system onto the eigenvectors, where the eigenvector associated with the largest eigenvalue is the axis of maximum variance, the eigenvector associated with the second largest eigenvalue is the orthogonal axis with the second largest variance, etc. Typically, only the N eigenvectors associated with the largest eigenvalues are used to define the subspace, where N is the desired subspace dimensionality.

In eigenspace terminology, each face image is projected by the top significant eigenvectors to obtain weights which are the best linearly weight the eigenfaces into a representation of the original image. Knowing the weights of the training images and a new test face image, a nearest neighbor approach determines the identity of the face.

2.2.2 Linear Discriminant Analysis

Linear Discriminant Analysis is a statistical approach for classifying samples of unknown classes based on training samples with known classes [11]. This technique aims to maximum between class (across users) variance and minimum within class (within user) variance. In these techniques a block represents a class, and there are a large variations between blocks but little variations within classes.

It searches for those vectors in underlying space that best discriminate among classes (rather than those that best describe the data). More formally given a number of independent features relative to which the data is described. LDA creates a linear combination of these which yields the largest mean difference between desire classes. Mathematically two measures are defined

- One is called within-class scatter matrix which is given by-

$$S_w = \sum_{j=1}^c \sum_{i=1}^{N_j} (X_i^j - \mu_j) (X_i^j - \mu_j)^T$$

where X_i^j is the i^{th} sample of class j , μ_j is mean of class j , c is number of classes, and N_j is number of samples in class j .

- Other is called between class scatter matrixes.

$$S_b = \sum_{j=1}^c (\mu_j - \mu) (\mu_j - \mu)^T$$

where μ represents the mean of all classes. The goal is to maximize the between class measure while minimizing the within class measure. To do this we maximize ratio $\frac{\det |S_b|}{\det |S_w|}$ to prove that if S_w is non-singular matrix then this ration is maximized when the column vectors of the projection matrix, W , are eigenvectors of $S_w^{-1} S_b$. It is noted that

- There are at most $c-1$ non zero generalized eigenvectors, and so an upper bound of f is $c-1$
- Require at least $c + t$ samples to guarantee that S_w does not become singular.

To solve this proposes the use of an intermediate space [12]. In both cases this intermediate space is chosen to be the PCA space. Thus the original t-dimensional space is projected onto an intermediate g-dimension space using PCA and then final f-dimension space LDA.

3. Existing System

In paper [13], the authors propose an efficient tracking algorithm based on rough and fine structure local sparse model. The proposed method in Fig. 4 utilizes the local appearance uniformity while the global appearance changes over time. An image patch in a fixed-space layout in a target area is extracted and encoded using a dictionary made up of patches of a plurality of target templates having sparsity constraints. The coding coefficients of the blocks spanning multiple templates are integrated by averaging and alignment pooling to obtain a robust representation of the target object. This operation helps to locate the object accurately and efficiently handles the partial occlusion by exploring the consistent parts of the object in the sequence of images.

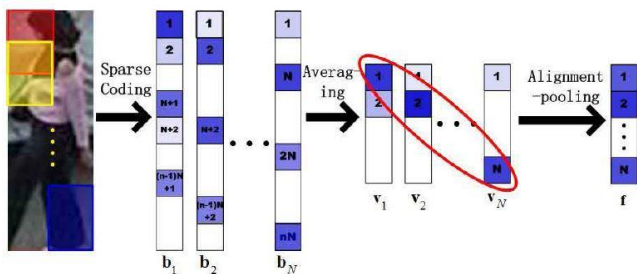


Figure 4: Illustration of local patches and feature vectors formed by averaging and alignment pooling

To make the representation more unique and robust, the likelihood of candidate regions is calculated here based on a combination of patches extracted with a coarse and fine strategy. The dictionary for local sparse coding is generated from a collection of collected templates that are sequentially updated based on the incremental subspace learning method [6]. A scheme of detecting the occlusion portion is also introduced to update the template more accurately without the occlusion pixels. The update module facilitates the proposed tracker account for target appearance changes caused by posture changes and lighting changes. In this paper, firstly, the sparse code of the local region is calculated by the average and alignment pool to simulate the appearance of the object for visual tracking. A new algorithm for constructing coarse and fine dictionaries is proposed for robust representation. Secondly, a template updating scheme based on incremental subspace learning is proposed to describe the appearance change of objects. The template update module is equipped with an occlusion detection module to include pixels belonging to the foreground object. Third, extensive experiments were conducted on a large baseline data set to evaluate the performance of the proposed algorithm for prior art approaches.

3.1 Screenshot

Fig. 5 shows that the tracking will continue even though the object undergoes any occlusion. Occlusion handling is done by the sparse coding method.



Figure 5: Tracking result at the time of occlusion

4. Problem Statement

Two major problems have been identified:

- The object can only be tracked if it does not move beyond the searched region.
- Various factors such as lighting and occlusions can affect the appearance of the target, thus making accurate tracking difficult.

To solve the first problem, we can attempt predicting the location of the target, and searching in a region centered on that location. But in making the prediction, it is necessary to consider the second problem as well. The previously obtained location measurements are likely not accurate, so the prediction method needs to be robust enough to handle this source of error. This report will discuss how these problems can be addressed with the Kalman Filter, and studies how well they are solved.

Next, the problem to be solved is detection of faces in an image. A human can do this easily, but a computer needs precise instructions and constraints. To make the task more manageable, Viola-Jones requires full view frontal upright faces. Thus in order to be detected, the entire face must point towards the camera and should not be tilted to either side. While it seems these constraints could diminish the algorithm's utility somewhat, because the detection step is most often followed by a recognition step, in practice these limits on pose are quite acceptable.

5. Proposed System

Detecting a face is a computer technology which let us know the locations and sizes of human faces. This helps in getting the facial features and avoiding other objects and things. In the present situation human face perception is a biggest research area. It is basically about detecting a human face through some trained features. Here

face detection is preliminary step for many other applications such as face recognition, video surveillance etc. Once the target has been detected then it can be tracked along its path.

5.1 Viola Jones algorithm

This algorithm helps us detect features of a face in a particular frame of a video sequence. This is the first object detection framework which gives a competition to real time detection rates. Paul Viola and Michael Jones are the ones who introduced this algorithm. They made this algorithm mainly by the issue of face detection. There are four steps which have to be followed to detect a face.

5.1.1 Haar Features

Haar features are a kind of rectangular boxes which are black and white. Haar features are simple rectangular feature which is the difference of the sum of pixels of areas inside the rectangle. This rectangle can be at any position of the frame and can scale the image. This modified feature set is called 2-rectangle feature. Each feature type can indicate the existence or the absence of certain characteristics in the frame, such as edges or changes in texture. These haar features are applied to determine the facial features.

5.1.2 Integral Image

The integral image part is used to sum all the pixels of a particular box to its left and above ones. The four corner values of the area are to be calculated. This makes avoid summing of each pixel in the region. This integral image conversion process is introduced just to speed up the process in calculating pixels [14].

5.1.3 Adaboost

It is a process used to find out relevant and irrelevant features. It uses the weak classifiers and weights to form a strong classifier. It finds the single rectangular feature and threshold which is the best to separate the faces and non-faces in training examples in terms of weighted error. It firstly starts with uniform weights while training. Next it evaluates the weighted error for each feature and picks the best [15].

Reevaluate the examples where incorrect classifiers will have more weight and correct classifiers will have less weight. Finally the classifier will contain the combination of correct classifiers which are having less weight. To reduce the computational time non-faces are discarded.

5.1.4 Cascading

This step is introduced to speed up the process and give an accurate result. This step consists of several stages where each stage consists of a strong classifier. All features are grouped into several stages. It detects faces in the frame by sliding a window over a frame [16].

Just to understand the Fig. 6 when an input is given it checks for certain classifier in the first stage and then so on. But it is passed to the successive stage if and only if it satisfies the preceding stage classifier.

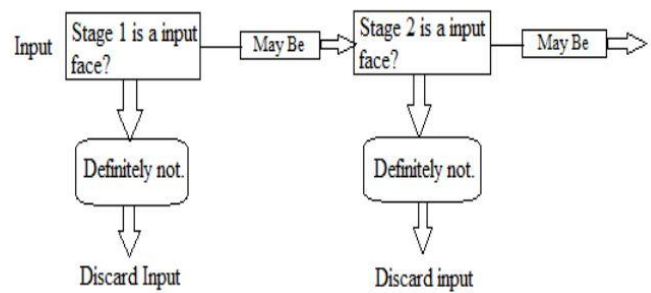


Figure 6: Cascading process

5.1.5 Implementation

This section describes an implementation of the system for face detection algorithm, which would enable us to detect faces in frames, forwarded from a video. Viola-Jones algorithm is used as the basis of the design. As we know there is some similarities in all human faces, here used this concept as a haar feature to detect face in image. Algorithm looks for specific haar feature of a face ,if these feature found algorithm pass the candidate to the next stage .Here the candidate is not whole image but just a rectangular part of this image known as sub window have a size of 24*24 pixel. With this window algorithm check whole image.

1) Pre-Processing

Before passing each sequential frame to face detecting algorithm the frame is processed to make the algorithm more efficient and more precise.

First, frame is converted from RGB format to GRAYSCALE. Then image size is reduced by performing pixels sampling. This helps the algorithm to work faster. In our implementation, the original image was scaled down by factor of 5. The last pre-processing step is aimed to improve precision of face detection algorithm. During this step the brightness is normalized and contrast is increased. As a result, the black and white features become significantly more noticeable.

2) Face Detection

After pre-processing the image, frame is forwarded to the next step of the algorithm, where different face poses are detected. If the face is found the coordinates of box containing the image are flipped back to correspond to actual image. When all of them finish to work, the resulting sets of faces are combined and provided as an input for the following post-processing step.

3) Post-Processing

After detection algorithm is applied we have a set of all detected faces with corresponding classifier type and coordinates of box containing the face.

5.2 Kalman Filter

Kalman filter technique is used to estimate the state of a linear system where state is assumed to be distributed by a Gaussian [17]. In 1960, R.E. Kalman [18] published his famous paper describing a recursive solution to the discrete-data linear filtering problem [18]. Fig. 7 shows the discrete

Kalman Filter cycle. Visual tracking is performed by predicting the target's position from the previous information and verifying the existence of the target at the predicted position.

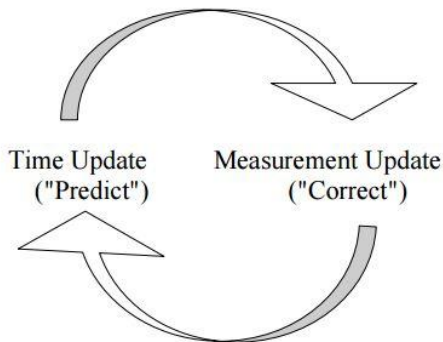


Figure 7: Discrete Kalman filter cycle

Secondly, the observed likelihood function and motion model must be learnt by some sample of image sequences before tracking is performed [19].

The Kalman filter is a set of mathematical equations that provides an efficient computational (recursive) means to estimate the state of a process in several aspects: it supports estimations of past, present, and even future states, and it can do the same even when the precise nature of the modeled system is unknown [20].

The Kalman filter estimates a process by using a form of feedback control. The filter estimates the process state at some time and then obtains feedback in the form of noisy measurements. The equations for Kalman filters fall in two groups: time update equations and measurement update equations. The time update equations are responsible for projecting forward (in time) the current state and error covariance estimates to obtain a priori estimate for the next time step. The measurement update equations are responsible for the feedback. That is used for incorporating a new measurement into the a priori estimate to obtain an improved a posteriori estimate. The time update equations can also be thought of as predictor equations, while the measurement update equations can be thought of as corrector equations. Fig. 7 shows the final estimation algorithm resembles that of a predictor-corrector algorithm for solving numerical problems. The time update projects the current state estimate ahead in time. The measurement update adjusts the projected estimate by an actual measurement at that time [20].

The time update projects the current state estimate ahead in time.

Time update equations [20] [21]:

State Prediction

$$X_{pred_k} = A * X_{k-1} + B * U_k + W_{k-1}$$

Error Covariance Prediction

$$P_{pred_k} = A * P_{k-1} * A^T + Q$$

X_{pred_k} is vector representing predicted process state at time k. X is a 4-dimensional vector [x y dx dy], where x and y

represent the coordinates of the target centre, and dx and dy represent its velocity.

X_{k-1} is vector representing process state at time k-1. A is a 4x4 process transition matrix of the form [21].

$$A = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

U_k is a control vector and B relates optional control vector U_k into state space. W_{k-1} is a process noise.

P_{pred_k} is predicted error covariance at time k. P_{k-1} is a matrix representing error covariance in the state prediction at time k-1, and Q is the process noise covariance [21].

Measurement update equations [20] [21]:

After predicting the state X_{pred_k} and its error covariance at time k using the time update steps, the Kalman filter next uses measurement to correct its prediction during the measurement update steps [21].

Kalman Gain

$$K_k = P_{pred_k} * H^T * (H * P_{pred_k} * H^T + R)^{-1}$$

State Update

$$X_k = X_{pred_k} + K_k * (Z_k - H * X_{pred_k})$$

Error Covariance Update

$$P_k = (I - K_k * H) * P_{pred_k}$$

K_k is Kalman gain. H is matrix converting state space into measurement space and R is measurement noise covariance. Determining R_k for set of measurement is difficult; many Kalman implementations statistically analyze training data to determine fixed R for all future time updates.

X_k is a process actual state. Using Kalman gain K_k and measurement Z_k process state X_k can be updated. Here Z_k is the most likely x and y coordinates of the targets in the frame. The final step in Kalman filter is to update the error covariance P_{pred_k} into P_k . After each time and measurement update pair, the process is repeated with previous posteriori estimates used to project or predict the new priori estimate [21].

5.2.1 Implementation

Tracking of target has been done using Kalman filter. Here tracking of any target can be done by providing the frame number from which tracking has to be started. From the selected frame any target can be picked for tracking by setting the position of the mask and then the target can be tracked in subsequent frames.

Following steps have been implemented for tracking a single target.

- Background frame has been calculated by taking average of all the pixels.

- Frame number has been selected from which tracking of any target has to be started.
- From selected frame target to be tracked has been selected by repositioning the mask.
- For selected target its centroid position has been found out and from centroid information all the equation of time and measurement update have been calculated. For selected frame the actual position X and error P has been calculated.
- For all remaining frames following steps have been repeated.

- 1) Background subtraction has been done to find out all the moving regions in the frame.
- 2) From the found regions, region with the lowest distance from the region selected in previous frame has been selected.
- 3) Selected region's centroid and other parameter have been used to calculate time and measurement update equations.
- 4) Obtained state position values X has been stored in Array for every frame.

Line joining each stored point has been drawn in every frame which shows the trajectory of the selected moving target.

6. Results

The Tracking algorithm has been successfully applied to a video. Results shows that the tracking algorithm using Kalman Filter has been able to track any moving objects in indoor as well as in outdoor environments and face detecting algorithm Viola-Jones can detect the face of the target object from that video. Here input is given as a video format or else it can take a live video with the help of a webcam. The video input should be given only in avi format.

The algorithms have been implemented and tested on Matlab R2014 a(64 bit) with operating system windows10. The system is having Pentium processor, 4 Gigabyte memory.

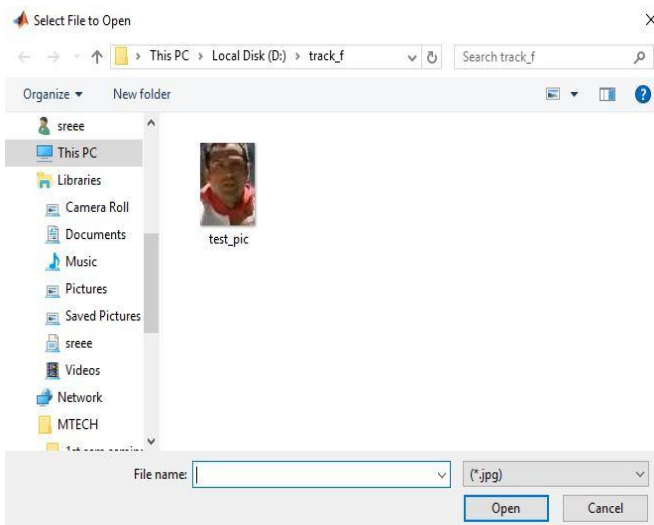


Figure 8: Picture to Select

Fig. 8 shows the target picture which is need to be selected for tracking.

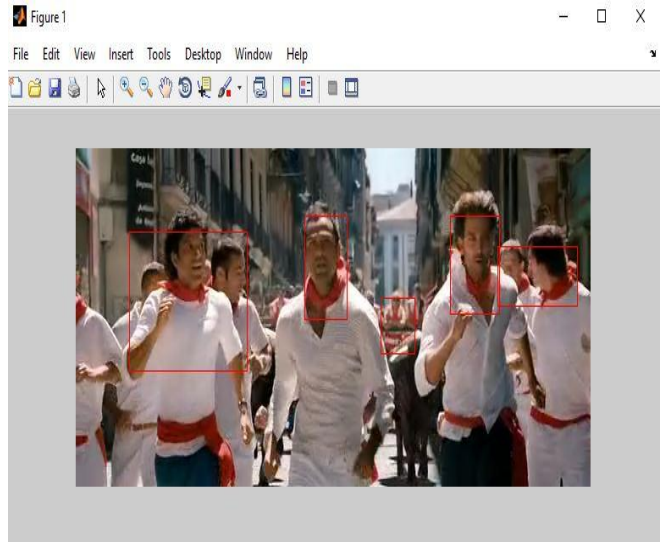


Figure 9: Multi-Person Detection

Multi-Person detection using Viola-Jones algorithm is shown in this Fig. 9.

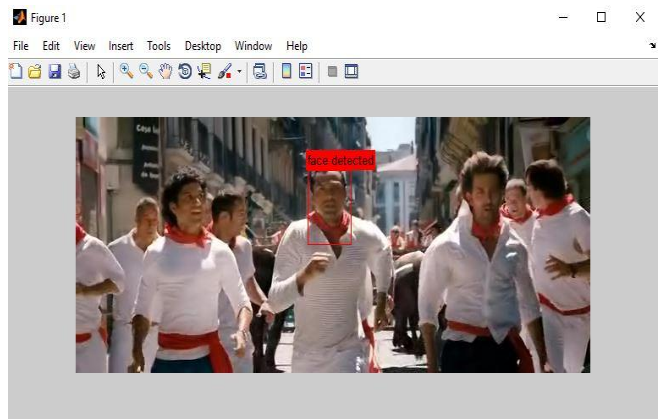


Figure 10: Target Person Detection

The face which is selected as the target person is identified using Viola-Jones algorithm is shown in this Fig. 10.

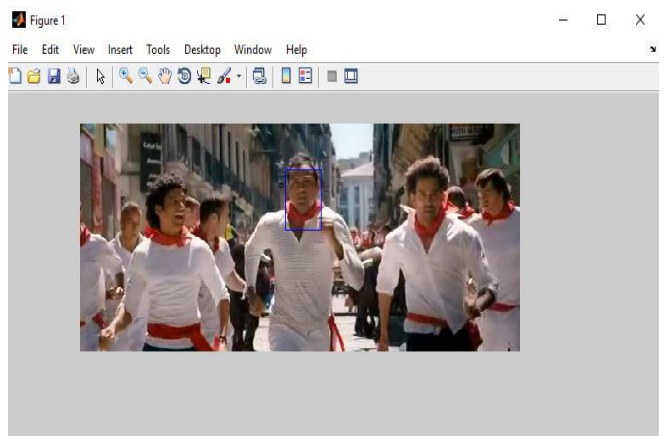


Figure 11: Target Tracking

Fig. 11 gives us the desired output. The face of the target person is detected using Viola-Jones algorithm and further tracking is done with the Kalman Filter algorithm.

7. Conclusion

This paper brings together new algorithms, representations, and insights which are quite generic and may well have broader application in computer vision and image processing. The approach presented here for face detection and target tracking decreases the computation time producing results with high accuracy. Tracking of a target in a video sequence is done using KF algorithm whereas Viola Jones is used for detecting facial features. Not only in video sequences, has it also been tested on live video using a webcam. Using this system many security and surveillance systems can be developed and required object can be traced down easily. In the coming days these algorithms can be used to detect a particular object rather than faces.

Here presented an approach for object detection which minimizes computation time while achieving high detection accuracy. The approach was used to construct a face detection system which is approximately 15 faster than any previous approach.

References

- [1] Comaniciu, D. and Ramesh, V. (2000). Robust detection and tracking of human faces with an active camera. In *Visual Surveillance, 2000. Proceedings. Third IEEE International Workshop on*, pages 11–18. IEEE.
- [2] Kennedy, J. and Eberhart, R. (1992). Particle swarm optimization proceedings of the international conference on neural networks. Australia IEEE, 1992.
- [3] Goldberg, D. E. and Holland, J. H. (1988). Genetic algorithms and machine learning. *Machine learning*, 3(2):95–99.
- [4] Pugh, J. and Martinoli, A. (2007). Inspiring and modeling multi-robot search with particle swarm optimization. In *Swarm Intelligence Symposium, 2007. SIS2007. IEEE*, pages 332–339. IEEE.
- [5] Shi, Y. et al. (2001). Particle swarm optimization: developments, applications and resources. In *Evolutionary computation, 2001. Proceedings of the 2001 Congress on*, volume 1, pages 81–86. IEEE.
- [6] X. Mei and H. Ling, "Robust visual tracking using l_1 minimization," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Sep./Oct. 2009, pp. 1436–1443.
- [7] Belhumeur, P. N., Hespanha, J. P., and Kriegman, D. J. (1997). Eigen faces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on pattern analysis and machine intelligence*, 19(7):711–720.
- [8] Turk, M. and Pentland, A. (1991). Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1):71–86.
- [9] Sirovich, L. and Kirby, M. (1987). Low - dimensional procedure for the characterization of human faces. *Josaa*, 4(3):519–524.
- [10] Cardoso, J. - F. (1997). Infomax and maximum likelihood for blind source separation. *IEEE Signal processing letters*, 4(4): 112–114.
- [11] Martínez, A. M. and Kak, A. C. (2001). Pca versus lda. *IEEE transactions on pattern analysis and machine intelligence*, 23(2): 228–233.
- [12] Swets, D. L. and Weng, J. J. (1996). Using Discriminant eigen features for image retrieval. *IEEE Transactions on pattern analysis and machine intelligence*, 18(8): 831–836.
- [13] X. Jia, H. Lu and M. H. Yang, "Visual Tracking via Coarse and Fine Structural Local Sparse Appearance Models," in *IEEE Transactions on Image Processing*, vol. 25, no. 10, pp.4555–4564, Oct. 2016.
- [14] Mikhaylov, D., Samoylov, A., Minin, P., and Egorov, A. (2014). Face detection and tracking from image and statistics gathering. In *Signal-Image Technology and Internet-Based Systems (SITIS), 2014 Tenth International Conference on*, pages 37 – 42. IEEE.
- [15] Li, S. Z. and Zhang, Z. (2004). Float boosts learning and statistical face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9): 1112–1123.
- [16] Benedict, A. A. and Jayapal, J. (2013). Face tracking techniques in color images: A study and review. *International Journal of Engineering*, 2(12).
- [17] Yilmaz, A., Javed, O., and Shah, M. (2006). Object tracking: A survey. *Acm computing surveys (CSUR)*, 38(4): 13.
- [18] Kalman, R. E. et al. (1960). A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1): 35–45.
- [19] Watada, J., Musa, Z., Jain, L. C., and Fulcher, J. (2010). Human tracking: A state - of art survey. In *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, pages 454–463. Springer.
- [20] Patel, H. A. and Thakore, D. G. (2013). Moving object tracking using kalman filter. *International Journal of Computer Science and Mobile Computing*, 2(4): 326–332.
- [21] Xu, S. and Chang, A. (2014). Robust object tracking using kalman filters with dynamic covariance. Cornell University.

Author Profile



Sreelekha M received the B.Tech degree from the Department of Information Technology, MEA Engineering College, Perinthalmanna and doing M.Tech from the Department of Computer Science, MEA Engineering College, Perinthalmanna.



Sufyan P received the B. Tech degree in Computer Science from College of Kidangoor, Kottayam in 2013 and received the M.Tech degree in Computer Science from MES College of Engineering, Kuttippuram in 2015. He is currently working as an Asst. Professor in MEA Engineering College, Perinthalmanna