MULTIPLE OBJECT TRACKING BY DECISION MAKING USING MEMORYLESS STATE TRANSITIONS

A Dissertation Submitted In Partial Fulfillment Of The Requirements For The Award Of The Degree

Of

MASTER OF TECHNOLOGY

IN

SIGNAL PROCESSING AND DIGITAL DESIGN

Submitted By:

ADITYA TYAGI

2K16/SPD/02

Under The Supervision of

Dr. RAJESH ROHILLA

Department of Electronics and Communication Engineering

Delhi Technological University

(Formerly Delhi College of Engineering)

Bawana Road, Delhi-110042

(SESSION 2016-2018)

Delhi Technological University

(Formerly Delhi College of Engineering)

Bawana Road, Delhi-110042

DECLARATION

I Aditya Tyagi, 2K16/SPD/02 student of M.Tech (SPD), hereby declare that the project dissertation titled **“Multiple Object Tracking By Decision Making Using Memoryless State Transitions”** which is submitted by me to the Department of Electronics and Communication Engineering, Delhi Technological University, Delhi has been carried out by me under the guidance of Dr. Rajesh Rohilla, in Delhi Technological University, New Delhi.

This major project is a part of the degree of M.Tech in Signal Processing and Digital Design. This is an original work and has not been submitted for any other degree in any other university.

Place: Delhi **ADITYA TYAGI**

Date:

Delhi Technological University
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

CERTIFICATE

This is to certify that the dissertation entitled **“Multiple Object Tracking By Decision Making Using Memoryless State Transitions”** being submitted by **Aditya Tyagi,** 2K16/SPD/02 in the partial fulfilment of the requirement for the reward of the degree of “Master of Technology” in “Signal Processing and Digital Design”, Delhi Technological University (Formerly Delhi College of Engineering), is an authentic record of the candidate’s own work carried out by him under my guidance. The information and data enclosed in this thesis is original and has not been submitted elsewhere for honoring of any other degree.

Place: Delhi
Date:

Dr. Rajesh Rohilla
SUPERVISOR
Associate Professor
Department of ECE
DTU
Delhi-110042

ACKNOWLEDGEMENT

I would like to express my heartily gratitude and thanks to my project advisor **Dr. Rajesh Rohilla**, **Associate Professor, Department of Electronics and Communication**, **Delhi Technological University, New Delhi** for his exemplary guidance, monitoring and constant encouragement in every stage of preparation of this work. I would also like to thank him for sparing the efforts in compiling the work presented here.

I would also like to thank my parents and friends for their support throughout the entire duration of the degree.

ADITYA TYAGI

2K16/SPD/02

ABSTRACT

This thesis work formulates tracking as decision making process where a tracker should follow an object despite ambiguous frames and having some limited computational budget. In tracking by detection, data association is main challenge i.e. to accurately associate noisy/ambiguous detected object of current frame to tracked objects from previous video frames and are linked to form trajectories of targets. In this work object tracking is done by making decisions regarding transition of memoryless states. The time for which an object is present in different video frames is modeled by memoryless states specifically known as Markov decision process(MDP). It has four different states: active, tracked, lost and inactive. Every new detected object enters active state. An active target can transition to tracked or inactive i.e. true positive from object detector should transition to tracked state, while a false alarm goes to inactive state. A tracked target will remain tracked, or transition to lost if the target is lost due to some reason, such as occlusion or disappearance from the field of view of the camera. Lost target can stay as lost if not viewed for some frames, or get back to tracked state if it appears again, or transition to inactive state if lost for very long time. Finally, inactive state is the terminal state for any target, i.e. an inactive target stays as inactive forever. This is for single object, likewise for tracking multiple objects several Markov decision processes are assembled. For tracking of object template, iterative Lucas-Kanade tracker is used which works by computing optical flow. Whenever the tracker fails to track the target due to change in appearance and the present state transitions to lost, only then the template is updated. History of previous templates are stored and the tracking template is the mean

of past templates from history of the tracked target. For the data association of tracked target and current detections, I have used determinative Hungarian algorithm and Murty’s best assignments. Similarity function needs to be learned which is equal to learning a policy for Markov decision process for data association. Policy determines which action to take for state transitioning. Reinforcement learning is used for policy learning that takes benefit from advantages of both offline and online learning in data association. Initially providing the ground truth trajectory of a target and similarity function, Markov decision process attempts to track the target and takes feedback from the ground truth. According to obtained feedback, decision process updates the similarity function to improve tracking. Similarity function is updated only when decision process makes a mistake in data association. Lastly, training is finished when Markov decision process can successfully track the target. This framework can handle the birth/death and appearance/disappearance of objects by treating them as state transitions. Also it is very robust in handling occlusions.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The purpose of object tracking is to determine the position of the object in video frames continuously against dynamic scenes i.e. to associate the target objects in consecutive frames. Videos are actually sequence of images, called as frames so all the image processing techniques can be applied only after dividing to individual frames. So object tracking is nothing but object recognition step in image processing. Tracking objects in videos is an important problem in computer vision which has attracted great attention.

It has various applications such as video surveillance, human computer interface, video indexing, sport competition, games(Kinect), traffic monitoring, motion recognition and autonomous driving etc. The goal of multi-object tracking (MOT) is to estimate the locations of multiple objects in the video and maintain their identities consistently in order to yield their individual trajectories. MOT is still a challenging problem, especially in crowded scenes with frequent occlusion, interaction among targets and so on.

1.1 CHALLENGES FACED

Multiple object tracking is a complex task and a challenging problem, incorporated with various difficulties such as maintaining object identities that are generated because of abrupt motion of object as well as the camera, change in appearance patterns of both the object as well as the background, complex object shapes, illumination changes, background clutter and occlusions.

Major challenge faced by multiple target tracking is the detection of object as it reappears in camera’s field of view i.e the association of tracked target to current detection and the problem is further exaggerated by the reality that object changes its appearance making the first frame appearance immaterial.

1.2 GOAL

Given an unconstrained video stream in which there are many target object moving in and out from the camera's field of view and also the object can change appearance significantly or can undergo partial or full occlusion. Main objective is to find object bounding box (position & area) and their trajectory and also maintaining identities of different object under occlusion i.e by correctly and accurately associating previously tracked target to object detections in current frame. When the video sequence is processed at the frame rate then this process will run for indefinitely longer time, this task is referred as long-term tracking problem.

1.3 MOTIVATION

This thesis work is majorly inspired by interactive occlusion handling i.e. robustly associating previously tracked targets to noisy object detections and aggressively learning putative appearance changes. On solving these tracking challenges, advanced applications in augmented reality, vehicle automation, healthcare, and security can be developed.

- Augmented reality: It is near to the actual reality. Basically digital information is integrated with the user environment in real time to produce more valuable results. By incorporating tracking, enhanced and enriched interaction can be achieved.
- Vehicle automation: Automated or self-driving cars require such robust tracking capabilities to handle occlusions (also maintain identities) of other vehicles or objects coming in the way of these automated cars.
- Healthcare: Gait analysis and body tracking have started to find more significant application in healthcare. Computer-vision tracking systems may demand that the moving bodies remain visibly separated or may demand particular orientations to a camera rather than a collaborating actor.

- Security and surveillance: Such systems continuously generate a large amount of video data. To analyze this huge data with human inspection is a tiresome work. So there is a requirement of automatic detection of threats and keep track on these objects when the reappear.
- Human-computer interaction: It provides an interface of computer with users through gesture recognition based on some hard-coded rules. Long-term trackers provide an advantage that one can imagine of being a personalized controller using gestures or using objects selected at the runtime.
- Object-centric stabilization: In hand-held camera where the user selects any arbitrary object to automatically adjust the camera settings, proposed tracking will enable user to restart the stabilization when object reappears in field of view of camera. Utility of this application is when observing distant object through digital zoom known as the auto focus problem.

Proposed tracking by decision making with robust data association capabilities can be incorporated to solve or reduce the problems defined above and is of great interest.

1.4 CONTRIBUTION

This thesis work mainly contributes by viewing long term tracking problem as decision making where the overall time for which an object is present in a video sequence is modeled by memoryless states specifically known as markov decision process. It has four different states: active, tracked, lost and inactive. It’s working is briefly explained: every new detected object enters active state. An active target can transition to tracked or Inactive i.e true positive from object detector should transition to tracked state, whereas false alarm should go to inactive state. Tracked target will remain tracked or transition to lost if the target is lost due to some reason such as partial/full occlusion or disappearance from the field of view of the camera. Lost target can stay as lost if not appeared for some frames or revert back to tracked state if it appears again or transition to inactive state if lost for long time. Inactive state is the terminal state for any target i.e. an inactive target stays as inactive forever. For tracking multiple objects several markov decision processes are assembled for each object present in a video stream. For template tracking iterative Lucas-Kanade tracker is used which works by computing optical flow. Online trackers updates tracking template every time on tracking a target, therefore they are highly influenced by tracking errors and results in building up of error during the runtime (drift). So the updation takes place when the tracker fails to track the target due to change in appearance and the present state transitions to lost, only then the template is replaced by the associated detection. History of previous templates are stored and the tracking template is the mean of past templates from history of the tracked target.

Another contribution in this novel approach lies in the data association part of tracked target and current detections, I have used determinative Hungarian algorithm and Murty's best assignments for associating tracked object to current detections. For this distance matrix is computed for analysis which contains the cost of associating tracks to detections respectively and this matrix is given as input to association algorithm for further computation to produce the best assignment accordingly. Reinforcement learning is used for policy learning that takes benefit from advantages of both offline and online learning in data association. Initially providing the ground truth trajectory of a target and similarity function, markov decision process attempts to track the target and collects feedback from the ground truth. According to the feedback, the markov decision process updates the similarity function to improve tracking. The similarity function is updated only when the markov decision process makes a mistake in data association. Training is finished when target is successfully tracked. This novel framework handles birth/death and appearance/disappearance of objects by treating them as state transitions. Also it is very robust in handling occlusions.

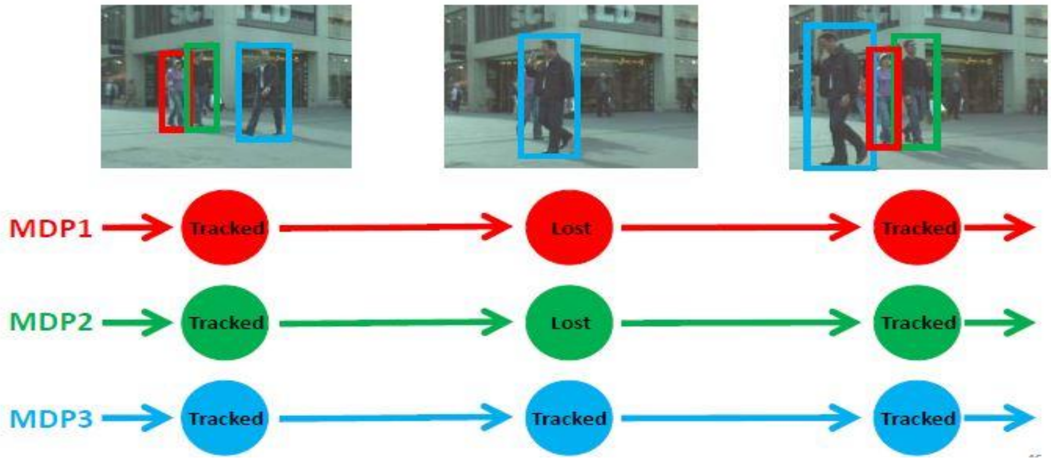


Figure 1.1: Tracking using decision making. 3 objects assigned with different decision processes undergoes state transitions accordingly.

CHAPTER 2

RELATED WORK

Tracker tracks the target object in every frame of a video sequence whereas detector localizes every object appearance that is observed so far and helps in reducing tracking drift by correcting whenever necessary. Machine learning in many times employed with both the approaches where the system uses learning to adapt with the changes and to learn a policy for decision making.

Tracker itself is modelled as an active agent that must make decisions to maximize its reward, which is basically correctness of track. Decisions ultimately specify where to devote finite computational resources at any point of time: should the agent process only a limited region around the currently predicted location i.e. track, or should search globally over the entire frame i.e. reinitialize, or should the agent use the predicted image region to update its appearance model for the object being tracked i.e. update, or should it be ignored, such decisions are very complicated when image evidence is ambiguous (due to partial occlusions). Instead of defining these decisions heuristically, data-driven techniques are ultimately used to learn good policies for active decision-making. This thesis work focuses on such decisions.

2.1 DECISIONS IN TRACKING

Decision regarding tracker reinitialization: Partial and/or full occlusions present major challenge, in that target's state cannot be reliably estimated while it's occluded. So, when a target first reappears, we must rely more on the current frame than our uncertain estimate of the target's previous state. Good single frame appearance models are crucial because they rapidly identify occlusion and re-appearance.

Decision regarding update of tracking template: On deciding when to update the appearance model, it has a big impact on tracker performance. On updating tracking template every time the tracker tracks a target, it is highly vulnerable to accumulate tracking drift. So in this work the tracking template is updated only when the tracker fails to track then template is replaced by associated detection and past k templates are stored in history. So the tracking template is the mean of past k templates stored in history.

2.2 MULTIPLE OBJECT TRACKING

It is most commonly viewed as a data association problem i.e. linking previous tracked object to one out of many detections. Trackers are tasked with grouping detections into coherent trajectories. In this work, the input is a set of spatiotemporal detections and associated appearance features, this assumes prior appearance and detection models. Such scenarios stand out in that the tracker is not provided with any initialization. Therefore, tracker must not only decide which detections belong to the same trajectory but also how many trajectories exist.

In multiple object tracking, maintaining identities of tracked targets is also very crucial task. Tracker which are aware of identities of multiple object requires labeling detections with identities rather than just clustering them. This problem requires some form of supervision at test time provided as initial bounding boxes for learning identity-specific appearance models.

2.3 MACHINE LEARNING

Traditionally, the object detectors are trained taking the assumption that all of the training examples are labelled. This assumption is very strong in our case since we want to train the detector from a single labelled example and a video stream. This problem can be formulated as a semi-supervised learning that exploits both the labelled and the unlabelled data. These methods typically assume independent and identically distributed data with certain properties, such as that the unlabelled examples form “natural” clusters in the feature space. Many algorithms based on similar assumptions are proposed, including the Expectation Maximization (EM), Self-learning, and Co-training. Expectation-Maximization is an old and basic method to find the estimates of model parameters in a given unlabelled data. EM is an iterative process which alternates between estimation of soft labels of the unlabelled data and training the classifier, in case of binary classification. EM technique was earlier successfully applied to document classification and learning of object categories. In semi-supervised learning terminology, the EM algorithm relies on the “low-density separation” assumption, which means that the classes are well separated. EM is sometimes interpreted as a “soft” version of self-learning.

Self-learning starts by training an initial classifier from a labelled training set; the classifier is then evaluated on the unlabelled data.

Co-training where one classifiers learns from the other.

There are many approaches derived through the combination of tracking-learning detection for example:

- Sometimes offline training of detector is done in order to get correct trajectory output of tracker and in case the trajectory is not correct an image search is performed on whole image to search the target.
- In some methods the detector is integrated with particle filter, particle filter does the tracking.

These methods are required to have offline training and the detector remains same throughout the runtime. To make the process more generalized online training approach is used where the real-time processing can be done.

CHAPTER 3

DECISION MAKING USING MDP

State transitions following markov property i.e. the effects of an action taken in a state depends only on that state and not on the prior history are specifically known as Markov Decision Process.

A Markov decision process (known as an MDP) is a discrete-time state transition system. It can be described technically with 4 components.

- Set of possible world states S
- Set of possible actions A
- Real valued reward function R(s, a)
- State transition function T: S x A -> S

3.1 MDP FRAMEWORK

Starting with the explanation of four main components which when combined together make up decision process and if follows markovian property (i.e. memoryless states) then known as markov's decision process.

Set of states S:

These states will play the role of outcomes in the decision theoretic approach, as well as providing whatever information is necessary for choosing actions. For example a robot navigating through a building, the state might be the room it's in, or the (x,y) coordinates. For a factory controller, it might be the temperature and pressure in the boiler. Practically the main assumption is that the set of states is finite and not too big to enumerate in our computer.

In this thesis work there are total four states defined for efficiently tracking an object namely,

- Active state
- Tracked state
- Lost state
- Inactive state

So therefore, set S = S_{active} U S_{track} U S_{lost} U $S_{inactive}$ and the transitions between these states are memoryless i.e the next state depends solely on the present state and there is no dependency on past state. So for every new detected object, it enter active state. Active state has further two possibilities to go into tracked or inactive state. True positive goes to tracked while a false alarm goes to inactive state. If entered tracked state then it can remain tracked or transition to lost if object undergoes occlusion or is outside the camera view. Lost target can remain lost for some threshold time or if detected again goes back to tracked state or transition to inactive if not detected again for time longer than the threshold. Inactive state is the terminal state as depicted in figure 3.1

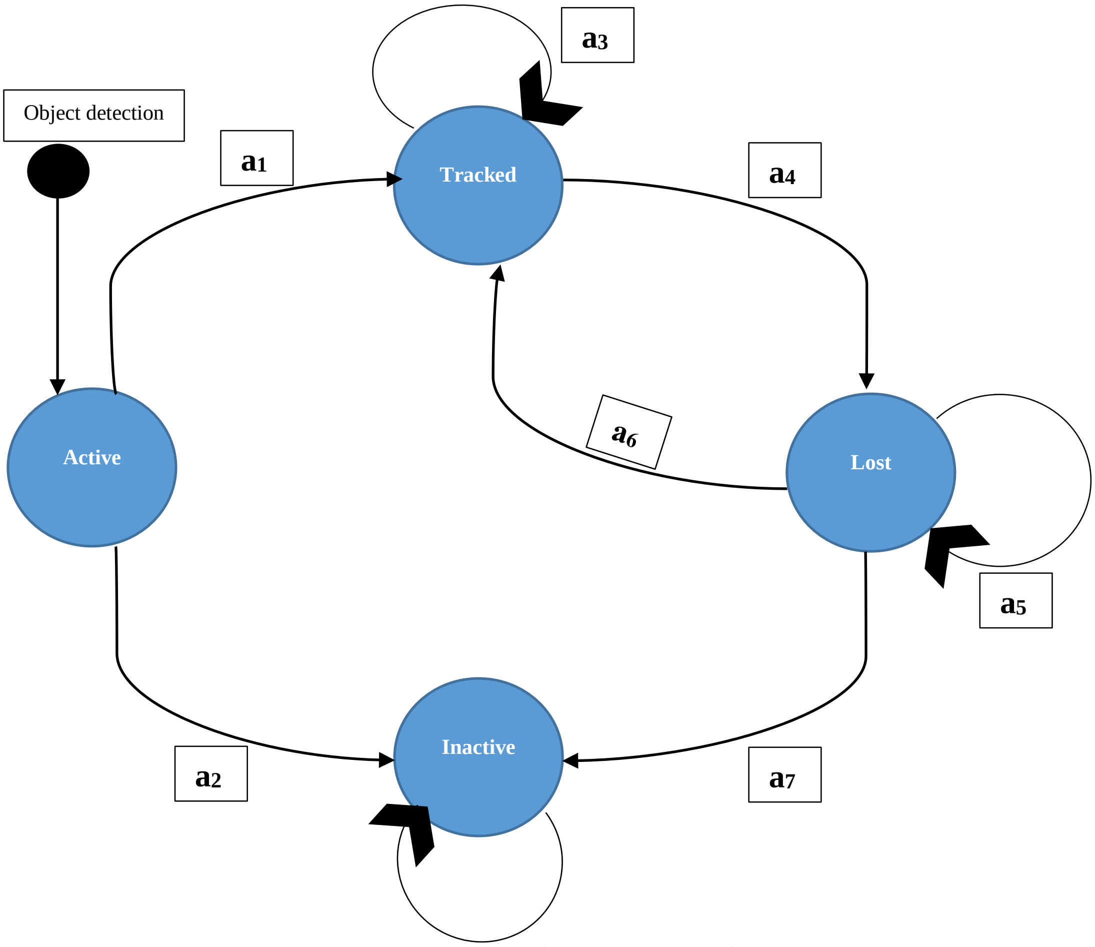


Figure 3.1: States and actions for MDP as defined in this work

Set of actions A:

These are chosen in the simple case, from a small finite set. There are two types of actions deterministic and stochastic. In case of former next state is defined for every action and present state but in the latter case probability distribution over next states is specified for each state and action. In this thesis work all the actions are deterministic and there are in total seven actions for seven possible transitions and are specified as follows:

a_1: action taken for transitioning from active to tracked state

a_2: action taken for transitioning from active to inactive state

a_3: action taken if the target will remain in tracked state

a_4: action taken for transitioning from tracked to lost state

a_5: action taken if the target will remain in lost state

a_6: action taken for transitioning from lost to tracked state

a_7: action taken for transitioning from lost to inactive state

State transition function T:

Transition function describes the dynamics of the world. They play the role of the next-state function in a problem solving search, except that every state is thought to be a possible consequence of taking an action in a state. So T is defined as T: S x A -> S i.e. for each state s_t and action a_t , the next state s_{t+1} will be defined.

In this thesis work there are seven possible transitions in total, e.g. executing action a_4 on a tracked target would transfer the target into a lost state, given as $T(s_{tracked} , a_4) = s_{lost}$

Reward function R:

Finally, there is a real-valued reward function on states. One can think of this as a short term utility function. R(s) tells that how good is it, from the agent perspective to be in state‘s’. Main goal is to find a policy (it determines which action to take) which maximizes the expected reward of executing that policy in the state.

Policy:

It determines the action which has to be taken out of set of actions. Classical planning results in a plan that is either ordered list of actions or partially ordered list of actions which is meant to execute without reference to state of environment. Conditional planning consists of building plans with branches in them, which observe something about state of the world and acts differently depending on the observation. In MDP, we assume that one could potentially go from one state to other state in one step. Therefore policy is computed. Policy is a mapping from states to actions and gives the best action to take, no matter at what state you happen to find yourself in.

Main challenge is to find criteria to determine good policy. Required criterion states that in the simplest case, try to find the policy that maximizes the expected reward of executing that policy in the state. It is particularly a very easy problem because it decomposes into a set of decision problems i.e for each state one should find the single action that maximizes expected reward.

3.2 POLICY AND REWARD FUCTION USED

For taking decisions using MDP, one should design policies in each state. All the defined policies are listed as follows:

Policy in active state:

Whenever a new object is detected then initialization occurs and it enters into active state and there are further two decisions i.e. transferring the true positive to tracked state or passing the false alarm to inactive. Basically it is preprocessing before tracking. Non-maximum suppression or thresholding detection scores like strategies are used. For deciding where to go, some basic classifier is required in which for some given feature vector input it classifies a detection into tracked or inactive target. 5D feature vector ϕ_{active} (s) is used as input which consists of 2D coordinates, score of detection, height and width and training videos gives training examples. Binary support vector machine is used as a classifier for classification.

For determining the best policy, reward function should be maximized and for that one should know the reward function which in this case is given by

$$R_{active}\,(s,a) = y(a)\;(\,w^{T}_{active}\,.\,\phi_{active}\,(s) + b_{active}\,)$$

where w_{active} and b_{active} are the weight and bias respectively and defines the hyperplane in support vector machine. y(a) can have values +1 or -1 acccordingly if action $a=a_1$ or $a=a_2$ respectively.

Policy in tracked state:

Whenever an object is in tracked state then formally there are further two decisive steps i.e. to keep it tracked or to move in lost state. If the target object does not undergo any occlusion or is present in camera field of view then it will keep on tracking otherwise transitions to lost. Appearance model for tracking the target object is to be build online. Tracking-Learning-Detection technique is used in this work. For defining policy in this state one should find forward backward error and for that optical flow should be calculated from sample points in a template to a new video frame. Given point $m=(m_x,m_y)$ in current frame I, and its respective location in new frame J is to be given by $n = m+f = (m_x+f_x , m_y+f_y)$ where $f = (f_x, f_y)$ is the optical flow at m. Stability of prediction is given by forward backward (FB) error. Now if prediction n is given, a new predict m’ can be found out by calculating backward flow of point n. FB error is given by Euclidean distance between m and m’ i.e the original point and forward backward predict:

$e(m)=||m - m'||^2$

Taking median of above,

$e_{medianFB}$= median ($\{e(m_i)\}_{i=1}^{n}$)

where n is number of points

Now tracking is unstable if this error is greater than the threshold value. It is not suitable to be decisive on tracking by considering optical flow alone. Therefore one has to decide this thing by analyzing other parameters also. As we know that appearance of false detections may not vary and also false detections like these are not detected consistently. So for this calculation of mean bounding box overlap is done, which is given by

o_{mean} = mean ($\{o(t_k, D_k)\}_{k=1}^{K}$)

where K is the number of past tracked frames

Reward function in tracked state s with feature representation

$\phi_{tracked}(s) = (e_{medianFB} , o_{mean})$ is given by

$$R_{tracked}(s, a) = \begin{cases} y(a) & if\ e_{medianFB} < e_0\ and\ o_{mean} > o_0 \\ -y(a) & otherwise \end{cases}$$

Where e_0 and o_0 are the defined thresholds having value 10 & 0.8 respectively and y(a) can have values +1 or -1 acccordingly if action $a=a_3$ or $a=a_4$.

So the target object will remain in tracked state if $e_{medianFB}$ is less than and o_{mean} is greater than some specified threshold.

Policy in lost state:

Whenever an object is in lost state then formally there are further three decisive steps i.e. to keep it in lost state or revert back to tracked or transition to inactive state. If target object has been lost for more than some specified threshold no. of frames then it transitions to inactive state. Otherwise data association is required for deciding between tracking again or to remain lost i.e a target gets transferred back to tracked state if it has been linked with any of the detections given by detector.

Let t denote lost object and d be an object detection. For predicting label $y \in \{+1, -1\}$ (having values y=+1 if tracked target is associated to one out of many detections or y=-1 if not associated) binary classifier with real valued linear function is used given by:

$f(t, d) = w^T . \phi(t, d) + b$

where w , b are classifier weight and bias that controls the function and ϕ(t, d) is basically the feature vector that gives similarity between tracked object and detection.

If f(t, d) ≥ 0 then y = +1 otherwise y = -1 and hence the reward function in lost state s with feature representation $\phi_{\text{lost}}(s) = \{\phi(t, d_k)\}_{k=1}^{M}$ is given by:

$$R_{\text{lost}}(s, a) = y(a)\left(\max_{k=1}^{M}\left(w^T.\phi(t, d_k) + b\right)\right)$$

Where k is the index of M deserving detections and y(a) can have values +1 or -1 acccordingly if action $a=a_6$ or $a=a_5$ respectively.

So policy is obtained by maximizing this reward function which can be done by learning w and b parameters.

3.3 PROCEDURE FOR TRAINING MDP

This thesis work assumes full observability i.e the system knows the new state which results from executing an action. Policy is obtained by maximizing the above defined reward functions which in turn determines the action that is going to be taken. So a complete policy is obtained for every specified value of weight and bias i.e. w & b parameters of the classifier and will take that action which maximize the defined reward function in given state.

Initially for training some random values of w and b is provided and a training set having no elements is initialized i.e $S_0 = \phi$. This thesis work uses MOT benchmark dataset for training and testing. So this procedure is applied on all the training videos following the policy obtained from initially specified parameters. Updation of w and b parameters is done only when there is a mistake in linking tracks with object detections.

Types of mistakes made are as follows:

1. Target object is linked to a wrong detection according to ground truth.

2. Target is not linked to any detection but according to ground truth there is an association present.

In type 1 mistake the $\phi(t_{ij}, d_k)$ is treated as negative example and stored in training set S. In type 2 mistake the $\phi(t_{ij}, d_k)$ is stored in training set S as positive example.

Using this updated set for retraining and likewise updation of binary classifier is done. A new policy is obtained after updating classifier that is used in next iteration. And this procedure runs iteratively till successful tracking of all objects.

CHAPTER 4

TECHNIQUE FOR TRACKING AND ASSOCIATION

There are various approaches for tracking and data association which was used by many researchers in the past. In this work TLD (Tracking-Learning-Detection) based tracker is used for tracking and combined Hungarian and Murty's best assignment approach is used for associating tracked targets to one out of many detections.

These techniques are further explained in this chapter.

4.1 TRACKING-LEARNING-DETECTION (TLD)

4.1.1 TLD FRAMEWORK

Tracking approximates the motion of object, it needs object initialization and results in smooth routes. But fails whenever object moves out of camera view i.e. when it disappears and results in building up of error during the runtime (drift).

Detection treats every frame of video stream to be independent and do the scanning operation on the full image to find the location of object and thus localizes the appearance. It does not result in drifting and does not fail when the object moves outside camera view. But they need offline training phase and method is not applicable to unknown objects. It results in two type of errors i.e. false positives and false negatives.

Neither tracking nor detection can single-handedly give solution to the object tracking problem. Learning method do the below mentioned works:

- Learning monitors the execution of both tracker and the detector.
- Learning approximates the errors of detector.
- Learning induces training examples in order to overcome the above errors in future.

Learning element is based on assumption that during the process both tracker and detector may fail. With the incorporation of learning in the algorithm, the detector can now generalise to more appearances of the objects and also can easily distinguish against the background. Main objective of learning is that when we are given with a single patch from the video, we need to concurrently learn the object classifier as well as make the correct labelling of patch as object or background.

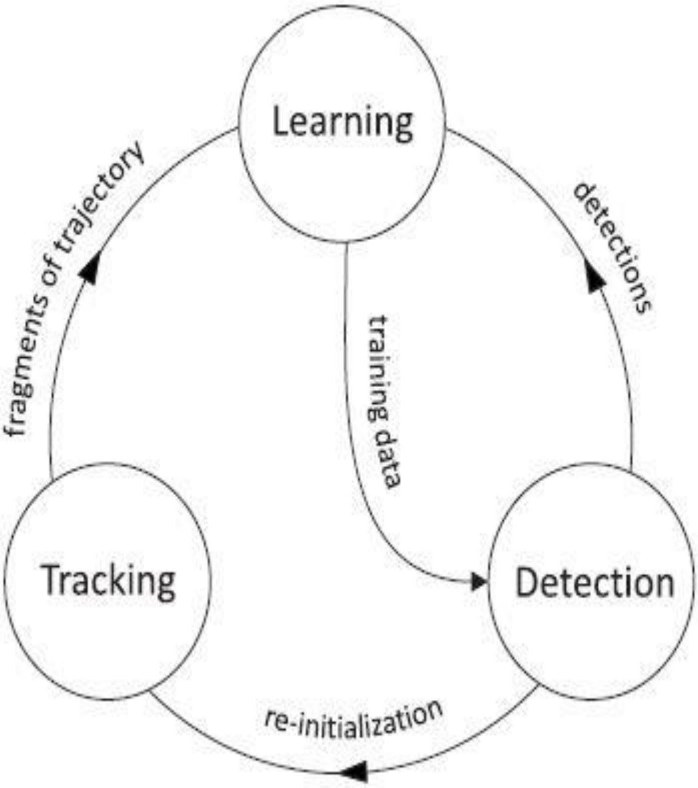


Figure 4.1: Diagram depicting TLD framework

4.1.2 PN LEARNING

Learning component of the TLD setup is explored out in this section. The new online learning paradigm known as P-N learning, is a semi-supervised learning for detection of objects from the video. Main objective of learning is that when we are given with a single patch from the video, we need to concurrently learn the object classifier as well as make the correct labelling of patch as object or background. The aim of the module is to enhance the efficiency of object detector by doing online process of the video sequence. This novel machine learning approach (PN learning) removes detector errors with the help of pair of ‘experts’: a) P-expert removes missed detections i.e. it identifies the false negatives b) N-experts removes false alarms i.e. it identifies the false positives. These experts make the errors themselves. However, their independence in making errors enables mutual cancellation of their errors which leads to sane and sensible learning. Its block diagram is shown in figure 4.2 below

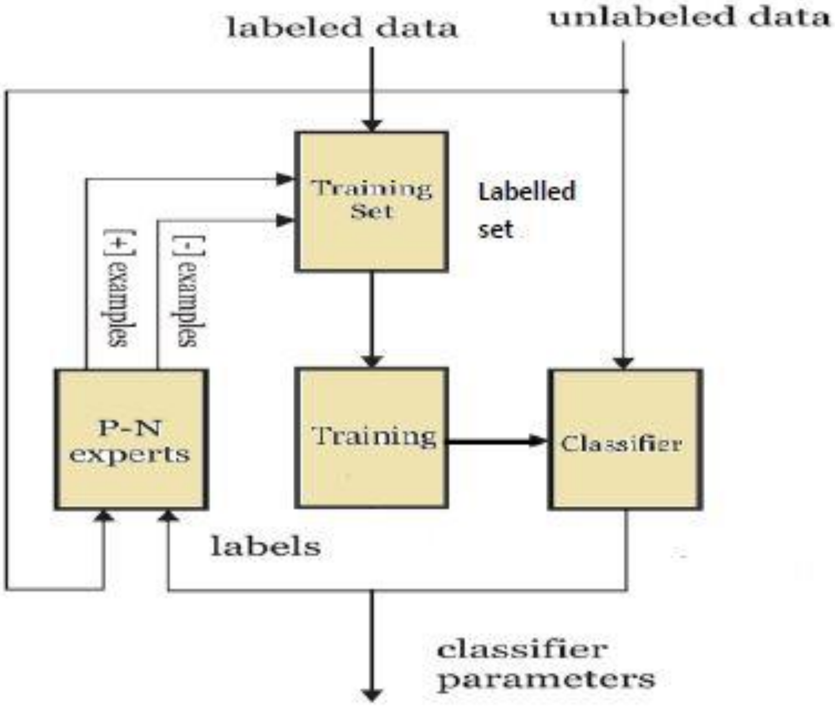


Figure 4.2: Block diagram depicting PN learning

4.1.3 TRACKER

This thesis work uses TLD approach having its tracking part implemented based on optical flow tracker. Target object is represented by a template which is detection bounding box or an image patch. For tracking, motion of target object is estimated between sequence of frames and for that optical flow of pixel points inside the target template is estimated. Given point $m=(m_x ,m_y)$ in current frame I, and its respective location $n = m+f = (m_x+f_x , m_y+f_y)$ in new frame J is computed by applying Lucas-Kanade method with pyramids iteratively, where $f=(f_x, f_y)$ is the optical flow at m. This method is based on gradients. Optical flow method recovers the motion in image patch that is template at every pixel which is translated from one point to another in future frames as shown in figure 4.3 below:

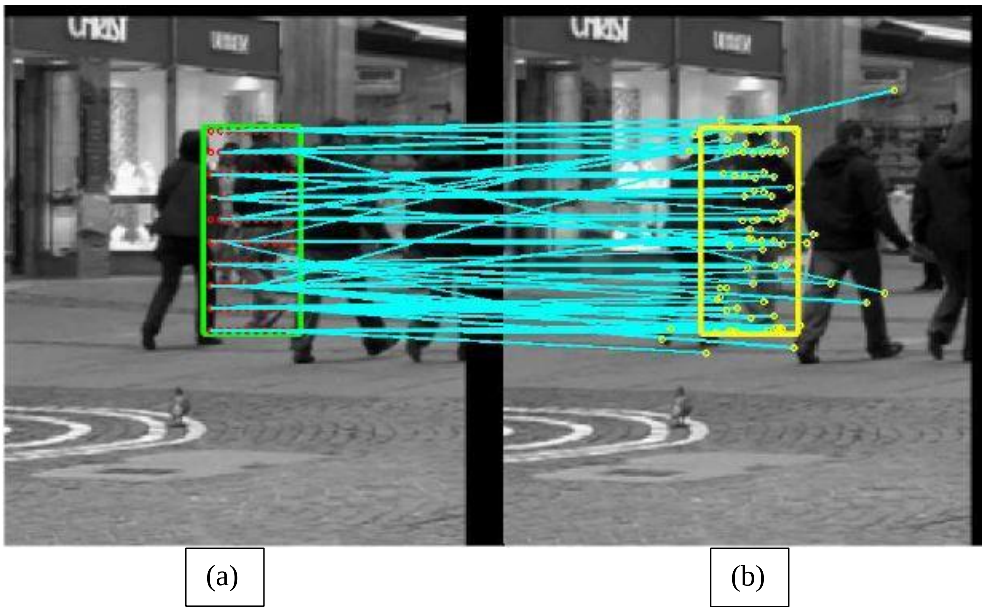


Figure 4.3: Template tracking by computing optical flow of points is shown using 2 consecutive image frames (a) & (b)

Lucas-Kanade technique assumes following constraints:

- Brightness consistency: image brightness in any small region will remain same only its location changes.
- Spatial coherence: in the neighbourhood of each pixel have similar motion as they belong to same surface.
- Temporal persistence: There is gradual motion of image patch over time.

Constraint equation of the optical flow method is:

$$I_x u + I_y v + I_t = 0 \qquad \text{where}$$

I_x, I_y, I_t are spatiotemporal derivatives of brightness in image u and v are horizontal and vertical optical flow respectively.

The image is divided into smaller parts and each part is assumed to move with constant velocity. Least-square fit is performed on constraint equation. Fitness is achieved by minimization of following equation:

$$\sum_{x\in\Omega} w^2[I_x u + I_y v + I_t]^2$$

W gives emphasis on the constraints at the center of every section. The solution to minimization is:

The LK tracker computes I_t with help of difference filter [-1, 1].

And u and v values are calculates as:

Compute I_x and I_y values by the kernel [-1 8 0 -8 1] / 12 and its transpose.

Compute It in between two images using [-1, 1] kernel.

Smoothening of the gradient components I_x, I_y, and I_t, by using separable and an isotropic kernel with effective 1-D coefficients as [1 4 6 4 1] / 16.

Solve above equation by 2 linear equation. The solution can be non-singular, singular or zero depending on eigen values. The eigen values of the matrix is compared with a threshold for noise and u and v values are calculated.

4.2 HUNGARIAN ALGORITHM

In multiple object tracking the main goal is to effectively link previously tracked object to detection in new frame. For this there is requirement of assignment algorithm. Solution to this problem is Hungarian algorithm also known as Munkres assignment algorithm which solves data association problem determinatively in O(n^3) time.

Cost matrix should be provided in order to obtain optimal assignments. Detailed stepwise algorithm is stated below:

1. Identify the smallest element of each row and subtract this entry from all the elements of its row.

2. Identify the smallest element of each column and subtract this entry from all the elements of its column.

3. Cover all the zero entries of the resulting matrix by drawing minimum number of lines through rows and columns.

4. Now there are two possible cases i.e
   (i) If minimum number of covering lines is n which is order of cost matrix, then assignment is optimal and terminate.
   (ii) If number of covering lines is less than n, then assignment is not yet optimal and further proceed to Step 5.

5. Find the smallest element not covered by any line and subtract it from each uncovered row then sum it with each covered column and revert back to step 3.

Above steps are clearly explained using an example as shown in figure 4.4 below:

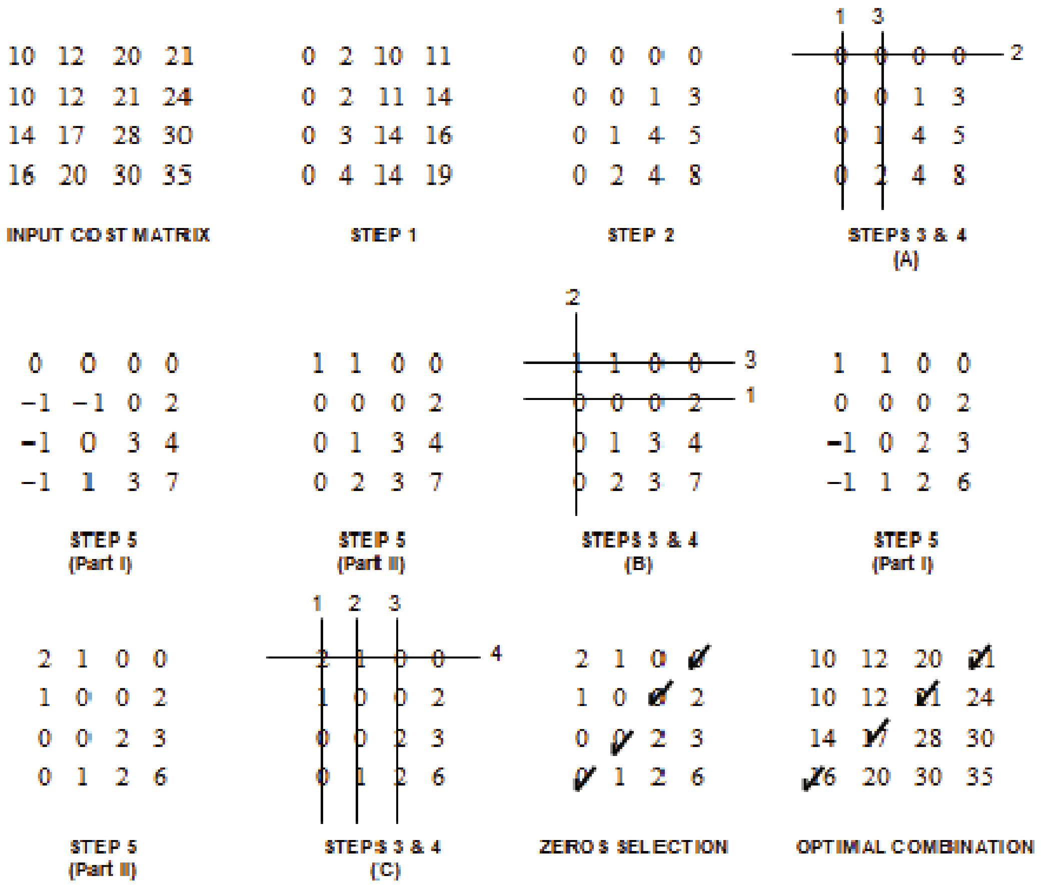


Figure 4.4: Various steps of Hungarian algorithm are shown

In this thesis work cost matrix is determined by calculating pairwise similarity between lost objects and detections that are not covered by tracked objects. To suppress the detections covered non maximum suppression approach is applied that depends on overlap of bounding boxes.

4.3 MURTY’S BEST ASSIGNMENTS

From Hungarian unique assignment is obtained but now if we want k best ranked assignments where k is user defined then Murty algorithm is used.

In practical scenario one’s work should be ready to incorporate randomness because when assigning tracks with detections there are not always crystal clear detections available but instead of that there are noisy and blurry detections as well. So in order to achieve better results, top k best assignments needs to be calculated and then choose randomly one out many best possible ranked assignments.

General algorithm is stated as follows:

1. Start with best assignment obtained by any association technique (in our case Hungarian algorithm is used)

2. Methodically tweak it by toggling matches in and out of the assignment

3. Create and store sorted list of best assignments so far

4. For every iterative sweep, toggle the matches in the next best assignment

5. K best assignments are found in decreasing order but only one per sweep.

Above stated steps are clearly explained using an example as shown in figure 4.5 below:

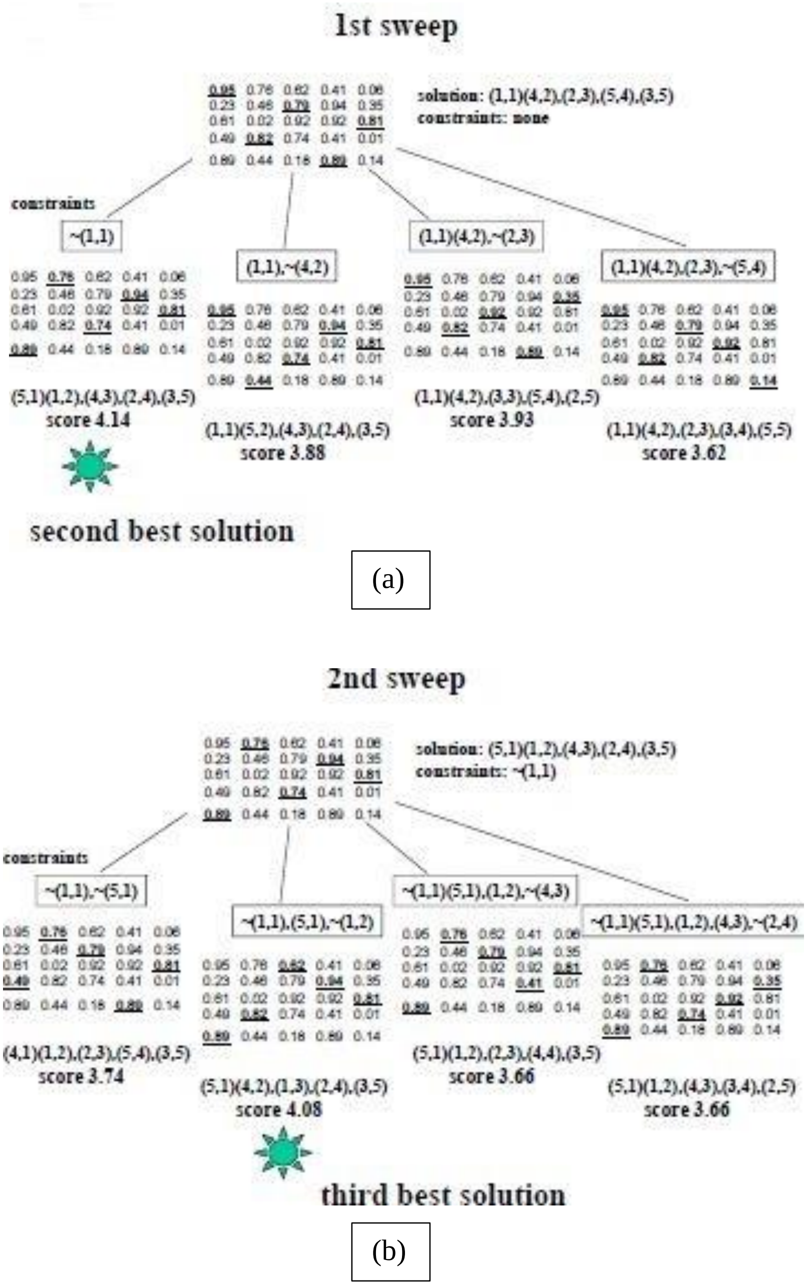


Figure 4.5: 1st and 2nd sweep of Murty’s algorithm is shown in (a) & (b) respectively

CHAPTER 5

IMPLEMENTATION OF PROPOSED WORK

This chapter further explain the implementation part regarding how to combine the above explained modules in order to achieve robust multiple object tracking by decision making using memoryless state transitions (specifically known as markov decision process) with tracker based on TLD approach and which associates data using Hungarian algorithm and further obtain ranked assignments using Murty's 'k' best assignment technique.

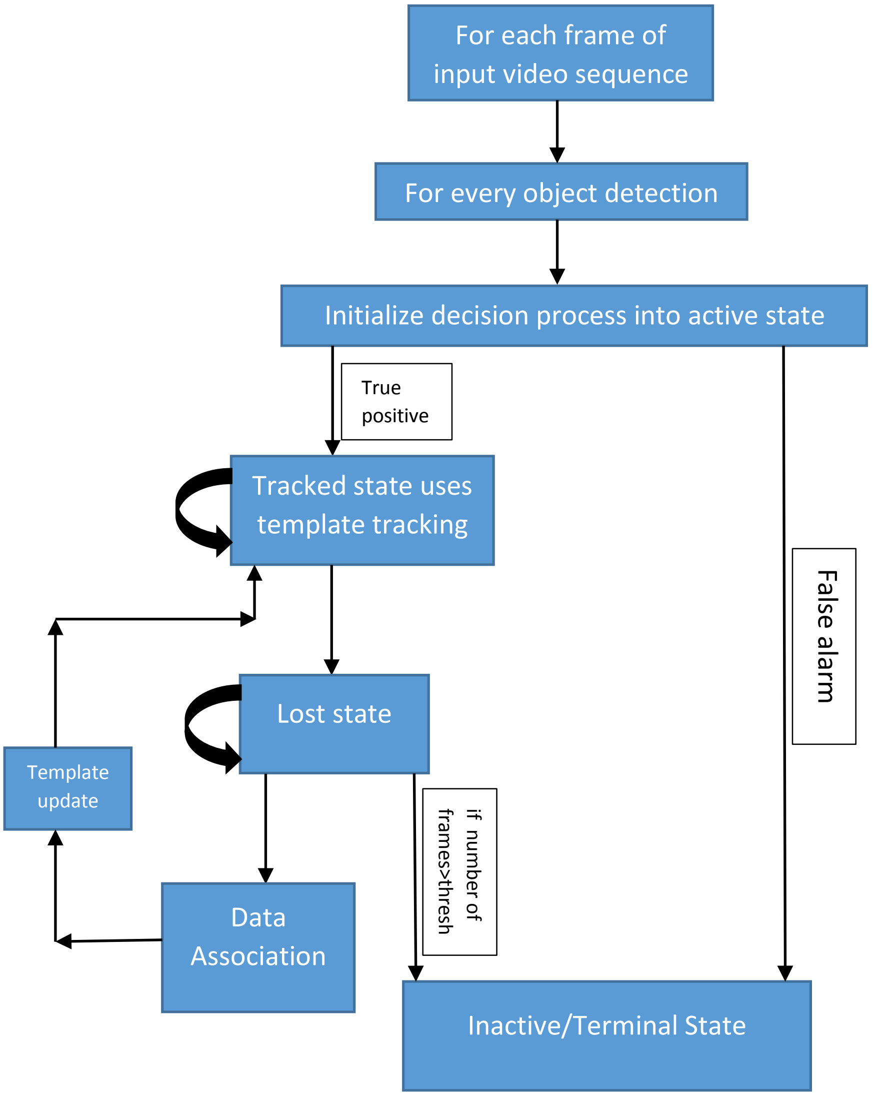

For each frame of input video sequence
For every object detection
Initialize decision process into active state
True positive
Tracked state uses template tracking
False alarm
Lost state
Template update
if number of frames>thresh
Data Association
Inactive/Terminal State

Figure 5.1: Block diagram of approach used in this work

5.1 PREREQUISITE KNOWLEDGE

Template representation is used to denote the appearance of object at any point of time and all the tracking is done using this template which is basically an image patch. Initialization of the target template with bounding box of detection is done where bounding box is made using top left corner coordinates, width and height. To measure the spatial similarity between two image patches ratio of intersection and union is measured and this is named overlap.

There are two approach for incorporating learning aspect in object tracking: offline learning and online learning. Basic difference between these two is that in offline approach supervision is required and based on it system learns a similarity function from ground truth for associating tracks to detections, it does not use history of target object in association whereas in online approach learning is performed in ongoing tracking process and produces positive and negative training samples which is used to learn similarity function for linking tracks to detection but in online case there are chances of tracking drift i.e if tracking results consists of some errors, then system will learn from these ambiguous training samples and produces somewhat wrong trajectories.

Also terms related to decision process such as state, action, transition function, reward function and policy is defined in chapter 3.

5.2 PROCEDURE

All the implementation and evaluation is done on sequences from MOT benchmark dataset. Here the time for which an object is present in different video frames is modeled by memoryless states specifically known as markov decision process. So for every object detection a new MDP is initialized and in chapter 3 it is defined that in this work there are 4 states namely active, tracked, lost and inactive and there are in total 7 actions which results in 7 different state transitions, also the transitions are completely memoryless which means next state is dependent only on present state and not on past history of states. These actions are a_1 , a_2 , a_3 , a_4 , a_5 , a_6 and a_7

Active state is initial state for every new detection and it can be transitioned to tracked or inactive based on action taken, and we know that policy gives the action for transformation of states and in order to determine best policy we need to maximize obtained rewards for that particular state. Also in chapter 3 reward function for active state is defined and is given by:

$$R_{active} (s, a) = y(a) (w^{T}_{active} . \phi_{active} (s) + b_{active})$$

Here

ϕ_{active} (s) is 5D feature vector which consists of 2D coordinates, score of detection, height and width

Also w_{active} and b_{active} are the weight and bias respectively and defines the hyperplane in support vector machine. y(a) can have values +1 or -1 acccordingly if action a=a_1 or a=a_2 respectively.

Now on learning this reward function and training is done on samples from training videos we obtain policy which in turn gives action that is to be taken. And let say after performing above steps true positive from detector is transferred to tracked state and false alarms goes to inactive as shown in figure 5.2

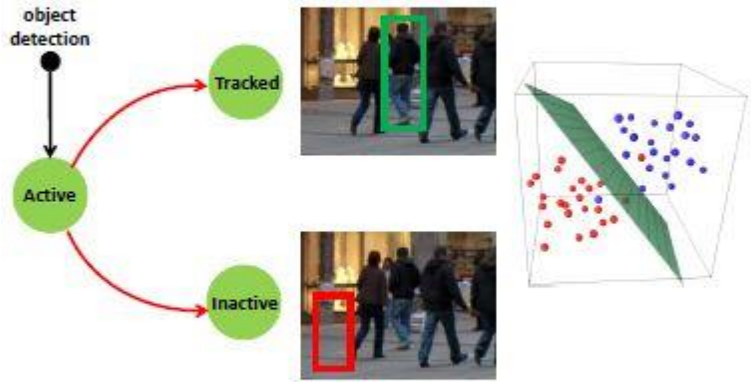


Figure 5.2: Decision boundary is shown for active state in which true and false detections are transferred to next state which is tracked or inactive accordingly.

In **tracked state** firstly tracking is performed, object appearance is represented by template that is used for tracking which is basically a small patch of target object. Initialization of template with bounding box is done in order to track. Tracker based on optical flow method is used in this work. Object moves from one position to another in consecutive image frames in a video stream. For tracking, motion of target object is estimated between sequence of frames and for that optical flow of pixel points inside the target template is estimated. Given point m = (m_x , m_y) in current frame I, and its location n = m+f = (m_x+f_x , m_y+f_y) in new frame J is computed by applying Lucas-Kanade method with pyramids iteratively, where f = (f_x, f_y) is the optical flow at m. This method is based on gradients. Optical flow method recovers the motion in image patch which is nothing but target template at every pixel which is translated from one point to another in consecutive frames. Further it has to choose from decision regarding transitioning to lost state or continue tracking in tracked state if object is successfully tracked. Stability of prediction is given by forward backward (FB) error. Now if prediction n is given, a new predict m' be found out by calculating backward flow of point n. FB error is given by Euclidean distance between m and m' i.e the original point and forward backward predict:

$$e(m) = ||m - m'||^2$$

Taking median of above,

$$e_{medianFB} = \text{median}\ (\{e(m_i)\}_{i=1}^{n})$$

where n is number of points

Now tracking is unstable if this error is greater than the threshold value. It is not suitable to be decisive on tracking by considering optical flow alone and one has to decide this thing by analyzing other parameters also. As we know that appearance of false detections may not vary e.g. background and also false detections like these are not detected consistently. So for this calculation of mean bounding box overlap is done, that is given by

$$o_{mean} = \text{mean}\ (\{o(t_k, D_k)\}_{k=1}^{K})$$

where K is the number of past tracked frames

Reward function in tracked state ‘s’ with feature representation

$\phi_{tracked}(s) = (e_{medianFB}, o_{mean})$ is given by:

$$R_{tracked}(s, a) = \begin{cases} y(a) & if\ e_{medianFB} < e_0\ and\ o_{mean} > o_0 \\ -y(a) & otherwise \end{cases}$$

Where e_0 and o_0 are the defined thresholds having values 10 & 0.8 respectively and y(a) can have values +1 or -1 accordingly if action $a=a_3$ or $a=a_4$.

So target object will remain in tracked state if $e_{medianFB}$ is less than and o_{mean} is greater than some specified threshold otherwise it is transferred to lost state due to occlusion or disappearance.

To compensate for appearance variation template needs to be updated and online trackers perform this updation whenever it tracks the target and as a result it is highly vulnerable to adopt errors during tracking and causes drift. So instead of updating everytime, the tracking template is updated only when tracker is unable to track and gets transitioned to lost state. Template is replaced by associated detection when it transitions back from lost to tracked and past K templates are stored in history of the object that is currently tracked. So the tracking template is the mean of past K templates stored in history.

In **lost state** there are three possible actions out of which one has to be chosen i.e. to keep it in lost state or revert back to tracked or transition to inactive state. If target object has been lost for more than some specified threshold no. of frames (which is 50 in this work) then it transitions to inactive state. Otherwise data association is required for deciding between tracking again or to remain lost i.e target gets transferred back to tracked state if it has been linked with any of the detections given by detector.

For **data association** part i have used Hungarian algorithm along with Murty’s best assignments (both the techniques are explained in chapter 4) to link previously tracked objects to one out of many detections given by detector. In this thesis work cost matrix is determined by calculating pairwise similarity between lost objects and detections that are not covered by tracked objects. To suppress the detections covered non maximum suppression approach is applied that depends on overlap of bounding boxes. Data association to one out of detections in lost state is shown in figure 5.3 below

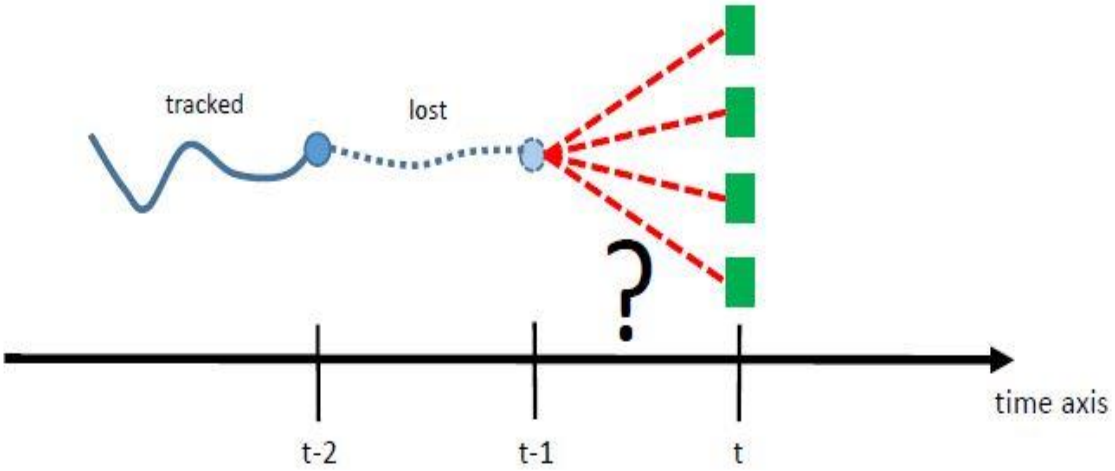


Figure 5.3: Shows data association in lost state where lost object is linked to one out of many detector output for current frame at time t

Matrix which contains cost for assigning track to detection based on similarity between lost objects and detections is given as input to data association technique for computing optimal assignment. And the optimal assignment matrix is given as input to Murty technique in order to obtain ranked assignments in descending order.

Let t denote lost object and d be detection. For predicting label $y \in \{+1, -1\}$ (having values $y = +1$ if tracked target is associated to one out of many detections or $y = -1$ if not associated) binary classifier with real value linear function is used given by:

$$f(t, d) = w^T . \phi(t, d) + b$$

where w, b are classifier weight and bias that controls the function and ϕ (t, d) is basically the feature vector that estimates similarity between tracked object and detection.

If $f(t, d) \geq 0$ then $y = +1$ otherwise $y = -1$ and hence the reward function in lost state 's' with feature representation $\phi_{lost}(s) = \{\phi(t, d_k)\}_{k=1}^{M}$ is given by:

$$R_{lost}(s, a) = y(a)\,(\max_{k=1}^{M} (w^T . \phi(t, d_k) + b))$$

Where 'k' is the index of M deserving detections and y(a) can have values +1 or -1 accordingly if action $a = a_6$ or $a = a_5$ respectively.

Hence policy is obtained by maximizing this reward function which can be done by learning w and b parameters. Training of classifier is done using **reinforcement learning**.

Given set of videos for training be $V=\{v_i\}_{i=1}^{N}$ where N is total number of sequences and let there be N_i ground truth targets $T_i = \{t_{ij}\}_{j=1}^{N_i}$ in video v_i. To track all the target objects, initially for training some random values of w and b is provided and a training set having no elements is initialized i.e $S_0 = \phi$. This thesis work uses MOT benchmark dataset for training and testing. So this procedure is applied on all the training videos following the policy obtained from initially specified parameters. Updation of w and b parameters is done only when there is a mistake in linking tracks with object detections. Let MDP is tracking jth target t_{ij} in lth frame of video v_i and is in lost state.

Types of mistakes made are as follows:

1. Target object is linked to a wrong detection according to ground truth.

2. Target is not linked to any detection but according to ground truth there is an association present.

In type1 mistake the ϕ(t_{ij}^{l}, d$_k$) is treated as negative example and stored in training set S. In type 2 mistake the ϕ(t_{ij}^{l}, d$_k$) is stored in training set S as positive example.

Using this updated set for retraining and likewise updation of binary classifier is done. Solve soft margin optimization problem to get max margin classifier for association using set S = $\{(\phi(t_k, d_k), y_k)\}_{k=1}^{M}$:

$$\min_{w,b,\xi} \frac{1}{2} \|w\|^2 + C \sum_{k=1}^{M} \xi_k$$

$$\text{such that} \qquad y_k(w^T \phi(t_k, d_k) + b) \geq 1 - \xi_k \,, \quad \xi_k \geq 0, \qquad \forall\, k$$

Here

ξ_k, k = 1,…,M denotes slack variables and ‘C’ is regularization parameter.

A new policy is obtained after updating classifier that is used in next iteration. And this procedure runs iteratively till successful tracking of all objects. Algorithm for reinforcement learning for associating data is given below:

Input: Videos V = $\{v_i\}_{i=1}^{N}$, ground truth trajectories T$_i$ = $\{t_{ij}\}_{j=1}^{N_i}$ and object detection D$_i$= $\{d_{ij}\}_{j=1}^{N_i'}$ for video v_i ; i=1,…..,N

Output: Binary classifier parameters w and b for data association

```
1. Initialize $w = w_0$ ; $b = b_0$ ; $S = \phi$
2. Repeat
   {
3. For every video $v_i$ in V
   {
4. For every target $t_{ij}$ in $v_i$
   {
           Start decision process in active state
           $l$ = index of first frame of $t_{ij}$ that is detected correct
           transition to tracked and set the appropriate target template
           **while** ($l \leq$ index of last frame of $t_{ij}$ )
           {
           Choose an action by following current policy
           Compute action $a_{gt}$ from ground truth
           **If**
             In lost state & $a \neq a_{gt}$
           **Then**
             Estimate label $y_k$ of pair $\phi(t_{ij}^{l}, d_k)$
             $S = S \cup \{(\phi(t_{ij}^{l}, d_k), y_k)\}$
             w, b = solution of soft margin problem
             break
           **Else**
             Execute action a
             $l = l+1$
           }
           **If** ( $l >$ index of last frame)  **Then** ($t_{ij}$ is successfully tracked)
   }
   }
5. }
   Until all target objects are tracked successfully
```

Here the used feature vector was denoted by ϕ(t, d) which consists of similarity between target object and detected object. As we store target history in form of K templates from previous K frames and then from each of these stored templates optical flow is calculated and bound its outcome in neighborhood of detected bounding box to compute feature and these feature are added that is grounded on how much the bounding box of target and detected object are similar. Type of feature stated below:

1. Forward-Backward(FB) error: It is already defined previously and is mean of median FB error from entire (denoted by ϕ_1), left side (denoted by ϕ_2), right side (denoted by ϕ_3), lower side (denoted by ϕ_4) and upper side (denoted by ϕ_5).
2. Normalized correlation coefficients: It gives two features ϕ_6 and ϕ_7 where mean of NCC b/w image patch around optical flow point which are matched is denoted by ϕ_6 and mean of NCC b/w patch of estimated bounding box(BB) using optical flow and the detection is denoted by ϕ_{7}.
3. Height ratio: It gives two features ϕ_8 and ϕ_9 where mean of ratios of height of estimated BB using optical flow and the detection is denoted by ϕ_8 and ratio in BB height of the target object and detection is given by ϕ_{9}.
4. Overlap: Its feature notation is ϕ_{10} which is mean of BB overlaps between detection and that BB which are estimated using optical flow.
5. Score: Its feature notation is ϕ_{11} that consists normalized score of detection.
6. Distance: After predicting motion of target object, find Euclidean distance b/w target object center and detected object center which is used as feature and is denoted by ϕ_{12}.

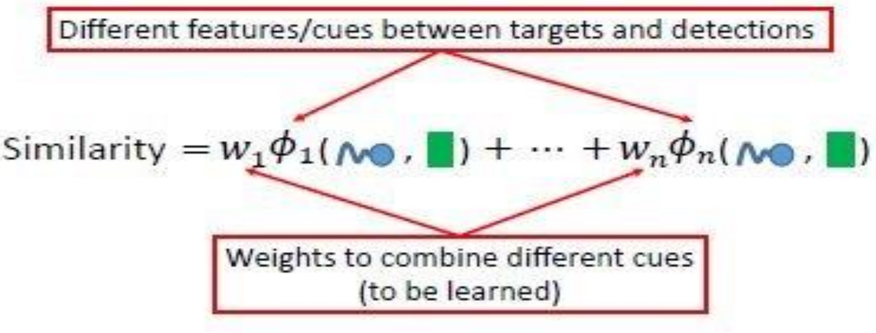

Different features/cues between targets and detections
Similarity $= w_1\Phi_1($, 🟩 $) + \cdots + w_n\Phi_n($, 🟩 $)$
Weights to combine different cues
(to be learned)

Figure 5.4: Depicts the similarity function which is to be learned for association

5.3 ALGORITHM FOR MOT

For tracking multiple objects that are present in frames of video sequence assignment of several markov decision processes is done for every object and it utilizes previously policy learned through above procedure. Processing of objects that are in tracked condition is done first for every new frame and after that for data association similarity measures are calculated between lost objects and detections that do not include the ones which are of objects in tracked condition/state. Similarity score computed by classifier are given to Hungarian algorithm and Murty’s best assignments technique for association. Non maxima suppression is used to suppres detections of tracked objects which lowers ambiguity in association. Lastly for every new object initialization of decision process is done. Algorithm is given below:

Input: video ‘v’, object detections D = $\{d_k\}_{k=1}^{N}$ and binary classifier parameters ‘w, b’ for association

Output: target trajectories T = $\{t_i\}_{i=1}^{M}$

For every video sequence:

```
1. Initialize T = $\phi$
2. For every frame l in v
   {
3. For every tracked object $t_i$ in T
        {
        Transition to next state of $t_i$ following policy
        }
4. For every lost object $t_i$ in T
        {
        For every detection $d_k$ which is not covered by tracked objects
        {
        calculate $f(t_i, d_k) = w^T . \phi(t_i, d_k) + b$
        }
        }
5.  Do data association using Hungarian and murty’s technique
6.  For every lost object $t_i$ in T
        {
        Transition to next state of $t_i$ following the assignments
        }
7.  For every new detections   that do not cover any tracked object in T
   {
        MDP initialization is done
        If
                action $a_1$ taken
        Then
                transition t to tracked state
        Else
                transition to inactive
   }
   }
   End
```

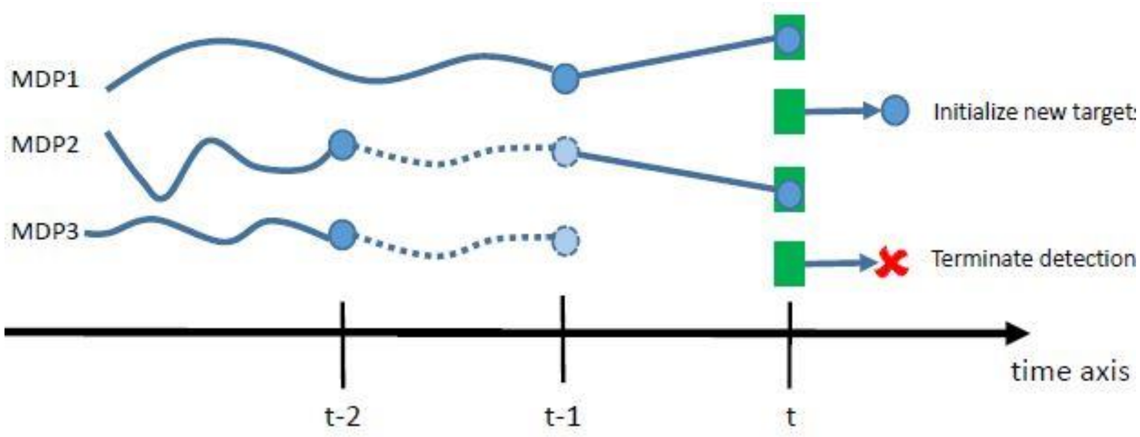


Figure 5.5: Shows separate decision process for tracking multiple objects in a video

CHAPTER 6

EXPERIMENTS AND RESULTS

In this chapter results obtained from experiments are showed and the performance parameters are explained. Tracking results thus obtained from decision making using memoryless state transitions are good based on evaluation.

Coding Environment: MATLAB R2018a

Operating system: Ubuntu

Dataset used: Multiple Object Tracking Benchmark

This dataset contains training and testing videos having eleven sequence each. It also consists of detection using Aggregate channel feature(ACF) object detector.

6.1 EVALUATION PARAMETERS

Various parameters used to evaluate results obtained are listed below:

1. Multiple object tracking accuracy: It describes all the different error such as misses, false positive and mismatch made by tracker. It gives a good estimate of performance at detecting object and maintaining their trajectories and is given by:

$$\text{MOTA} = 1 - \frac{\sum_t (m_t + fp_t + mm_t)}{\sum_t g_t}$$

   Here m_t, fp_t and mm_t are the number of misses, false positive and mismatches for time t.

2. Multiple object tracking precision: Ability to estimate exact object position precisely independent of object configuration. Basically it is the error in measured position for matched object hypothesis pair for all frames and is average over number of matches made and is given by:

$$\text{MOTP} = \frac{\sum_{i,t} d_t^i}{\sum_t c_t}$$

where i=1,….,n for n objects for every time frame t

3. Mostly tracked targets: It is given by the percentage of objects from ground truth whose covering of trajectory are atleast 80 percent by tracking output.

4. Mostly lost targets: It is given by the percentage of objects from ground truth whose covering of trajectory is less than 20 percent by tracking output.

5. Number of total false positives obtained

6. Number of total false negatives obtained

7. Total number of frames processed in 1 sec.

6.2 OBTAINED RESULTS

Firstly I have checked the results by dividing videos from training set into two parts, out of which one is used to train while other is used to test the system. Six out of eleven(Tud campus, adl rundle8, eth sunnyday, eth pedcross2, kitti17, venice2) is used for testing and remaining for training(Tud stadmitte, eth bahnhof, kitti13, adl rundle6). This is done to check the outcomes of the proposed system as ground truth annotations for test videos are not available in MOT15 dataset. And lastly the testing is performed on the test set and captured frames of resulting videos are shown in this section.

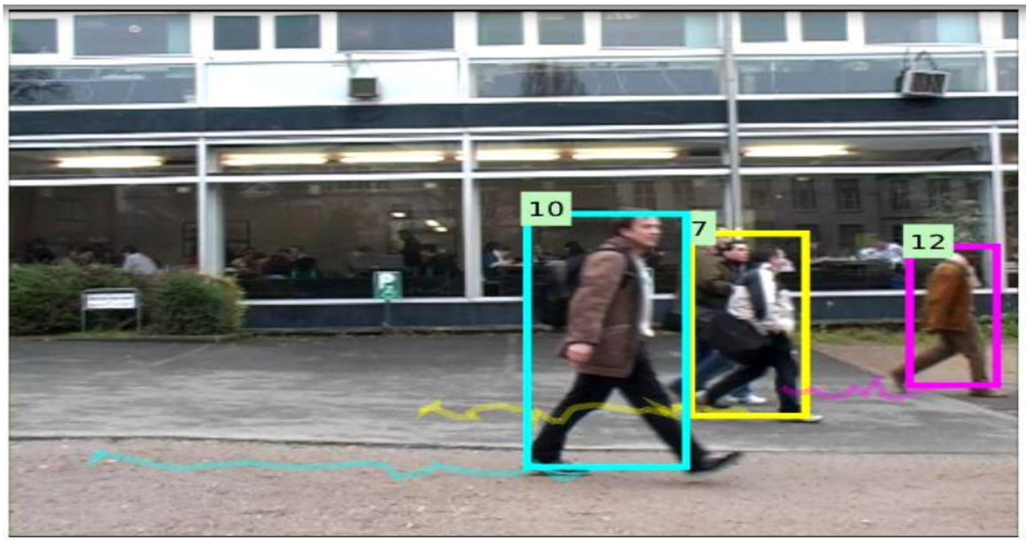
Description of video used for testing: Tud campus video is available in mot15 benchmark dataset under training sequences. It's background is stationary.

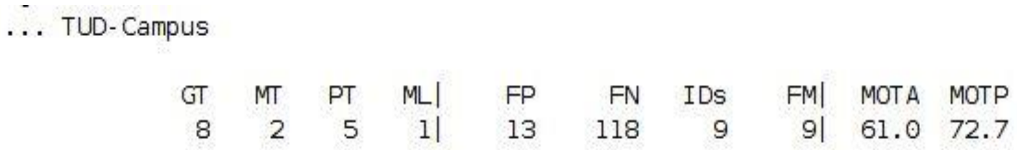
Pedestrians which are coming in field of view in different frames are tracked successfully and their trajectories are shown.

Dimensions of video:- 826 x 571

Frame rate:- 9

Length:- 00:00:07

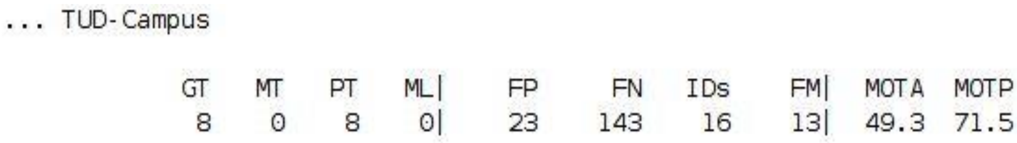




```
... TUD-Campus

        GT    MT    PT    ML|    FP     FN   IDs    FM|  MOTA  MOTP
         8     2     5     1|    13    118     9     9|  61.0  72.7
```

(a)



```
... TUD-Campus

        GT    MT    PT    ML|    FP     FN   IDs    FM|  MOTA  MOTP
         8     0     8     0|    23    143    16    13|  49.3  71.5
```

(b)

Figure 6.1: Results of tracking obtained from testing on TUD CAMPUS and training on TUD STADTMITTE (a) From proposed method (b) From previous method

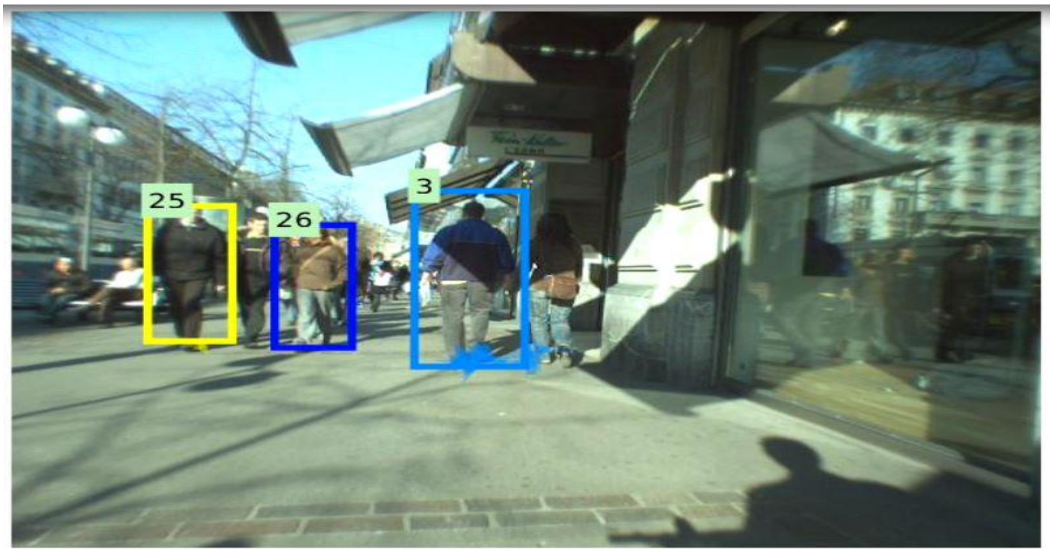
Description of video used for testing: Eth Sunnyday video is available in mot15 dataset for training but I have used it for testing to check proposed system functionality. It's background is not stationary i.e it keeps on changing as capturing camera is moving.

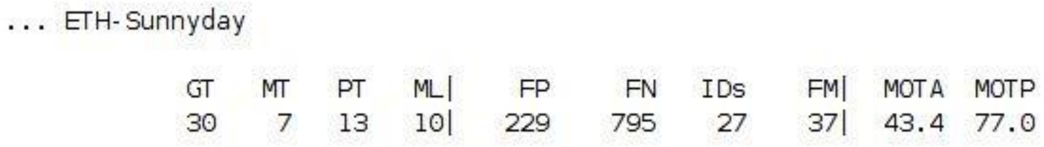
Pedestrians which are coming in field of view of recording camera are tracked.

Dimensions of video:- 826 x 571

Frame rate:- 9

Length:- 00:00:19

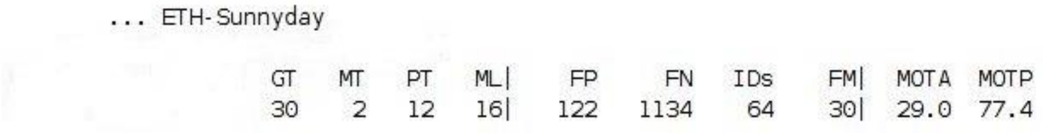




```
... ETH-Sunnyday

                  GT    MT    PT    ML|    FP     FN    IDs    FM|  MOTA  MOTP
                  30     7    13    10|   229    795     27    37|  43.4  77.0
```

(a)



```
... ETH-Sunnyday

                  GT   MT   PT   ML|    FP     FN   IDs    FM|  MOTA  MOTP
                  30    2   12   16|   122   1134    64    30|  29.0  77.4
```

(b)

Figure 6.2: Tracking result from testing on ETH SUNNYDAY and trained on ETH BAHNHOF (a) From proposed method (b) From previous method

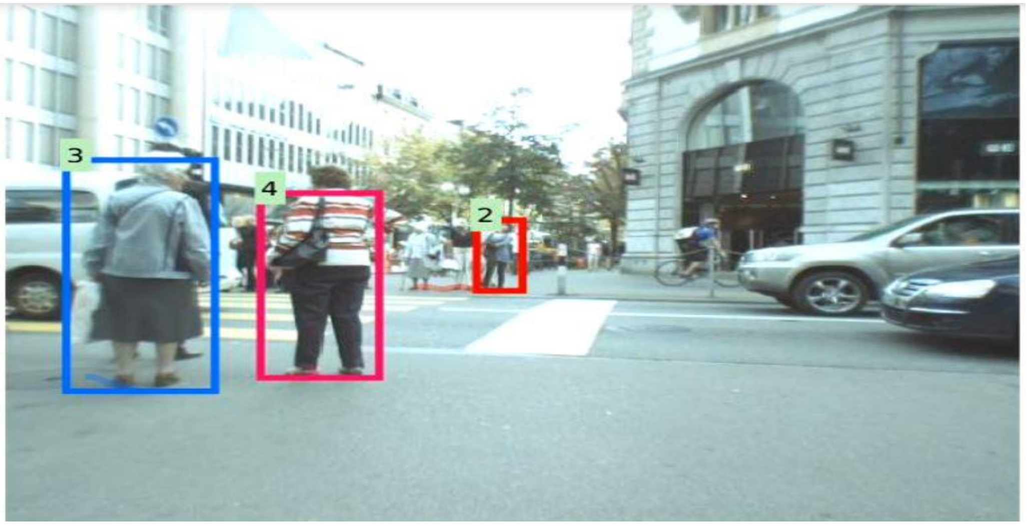
Description of video used for testing: Eth pedcross2 video is taken from mot15 benchmark dataset. It is having non stationary background i.e. keeps on changing as capturing camera is moving.

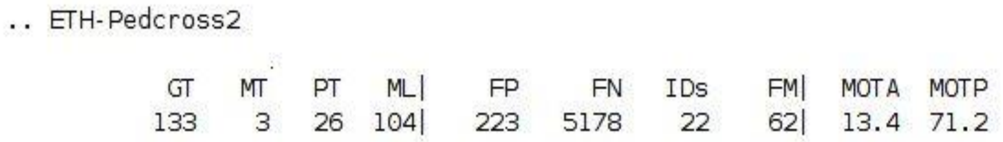
Most persons which are coming in field of view in sequence of frames are tracked.

Dimensions of video:- 836 x 601

Frame rate:- 9

Length:- 00:00:19

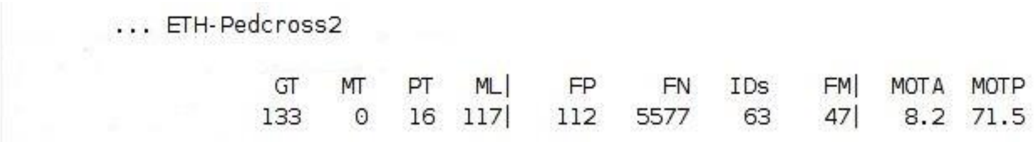




```
.. ETH-Pedcross2

      GT   MT   PT   ML|    FP     FN   IDs    FM|  MOTA  MOTP
     133    3   26  104|   223   5178    22    62|  13.4  71.2
```

(a)



```
... ETH-Pedcross2

      GT   MT   PT   ML|    FP     FN   IDs    FM|  MOTA  MOTP
     133    0   16  117|   112   5577    63    47|   8.2  71.5
```

(b)

Figure 6.3: Results obtained from testing on ETH PEDCROSS2 and training on ETH BAHNHOF (a) From proposed method (b) From previous method

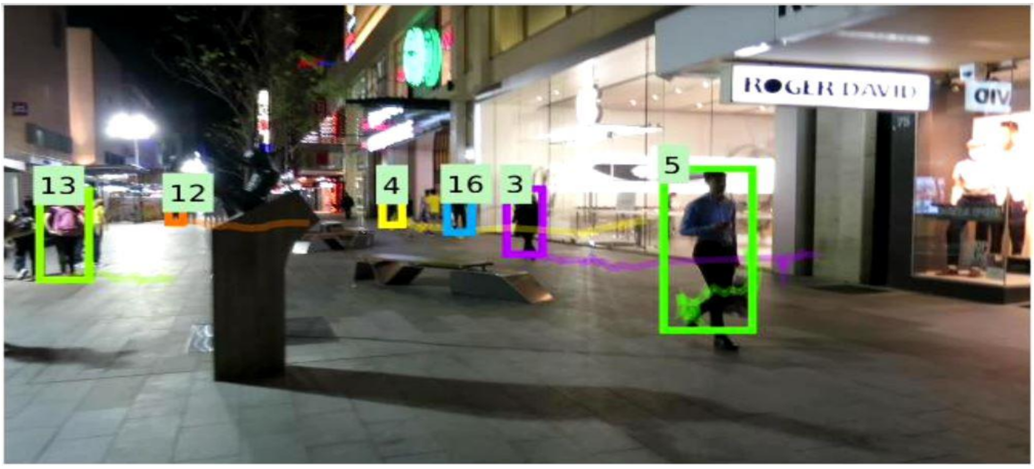
Description of video used for testing: Adl rundle8 video is available in mot15 benchmark dataset for training purpose. This is a high definition video with non stationary background which means it keeps on changing as camera through which this video is recorded is moving.

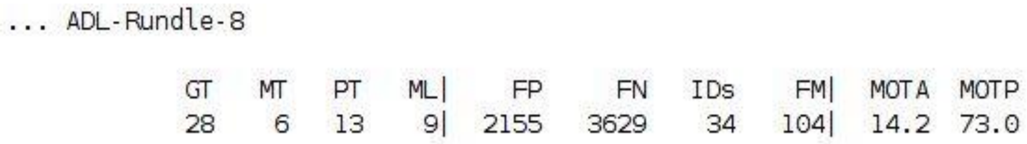
Most objects which are coming in capturing camera field of view are tracked.

Dimensions of video:- 1472 x 832

Frame rate:- 9

Length:- 00:00:19

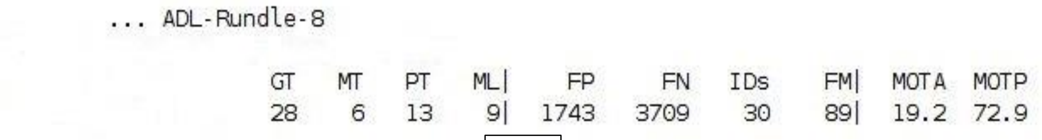




```
... ADL-Rundle-8

             GT    MT    PT    ML|     FP     FN   IDs     FM|  MOTA  MOTP
             28     6    13     9|   2155   3629    34   104|  14.2  73.0
```

(a)



```
... ADL-Rundle-8

        GT    MT    PT    ML|     FP     FN   IDs    FM|  MOTA  MOTP
        28     6    13     9|   1743   3709    30    89|  19.2  72.9
```

(b)

Figure 6.4: Results of testing on ADL RUNDLE 8 and training on ADL RUNDLE 6

(a) From proposed method (b) From previous method

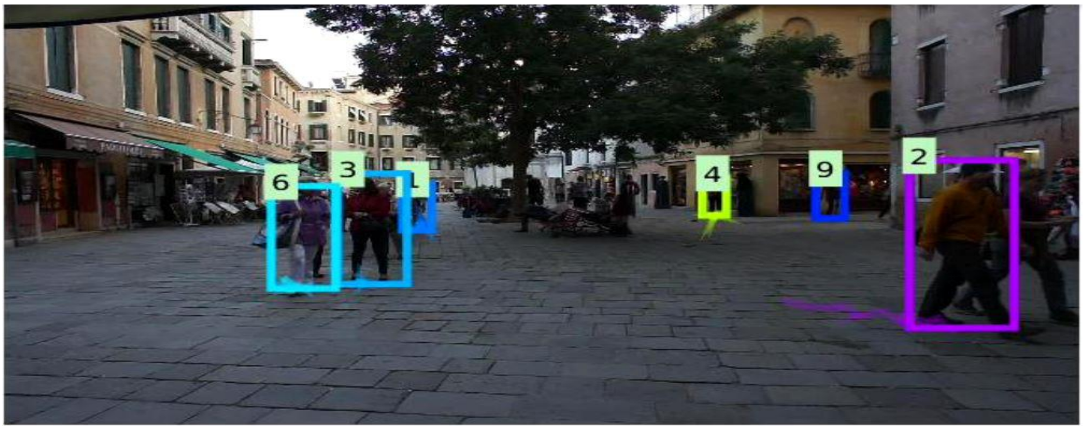
Description of video used for testing: Venice2 video is available in mot15 benchmark in training sequences. This is a high definition video and is taken from a stationary camera having stationary background.

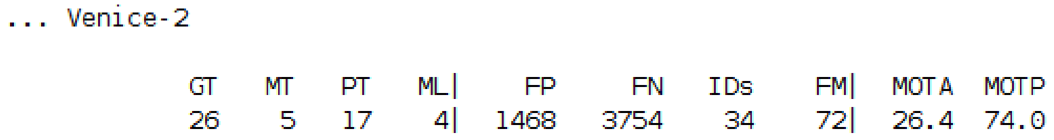
Most pedestrians which are coming in field of view in sequence of frames are tracked and also object occlusions are handled.

Video Resolution:- 1472 x 832

Frame rate:- 9

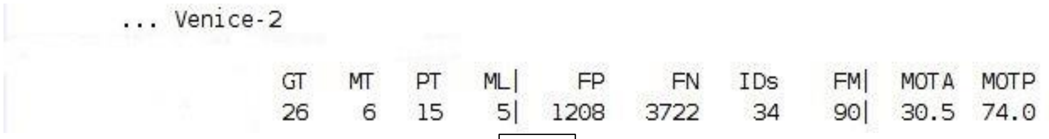
Length:- 00:01:06





```
... Venice-2
                      GT    MT    PT    ML|    FP     FN    IDs    FM|  MOTA  MOTP
                      26     5    17     4|  1468   3754     34    72|  26.4  74.0
```

(a)



```
         ... Venice-2
                          GT    MT    PT    ML|    FP     FN    IDs    FM|  MOTA  MOTP
                          26     6    15     5|  1208   3722     34    90|  30.5  74.0
```

(b)

Figure 6.5: Result that we get from testing on VENICE2 and training on ADL RUNDLE6
(a) From proposed method (b) From previous method

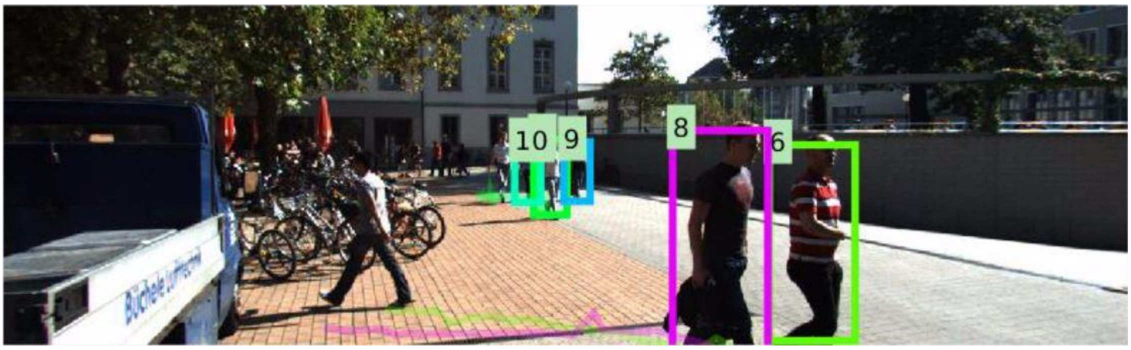
Description of video used for testing: Kitti 17 video is taken from mot benchmark dataset dataset for purpose of training with ground truth annotations. This is a high definition video having static background.

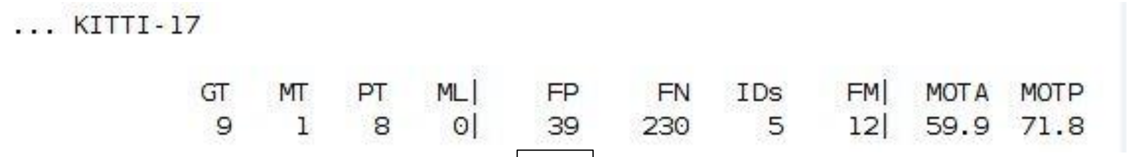
Most person that are coming in camera's view in different frames are tracked also their identities are maintained as long as possible.

Video Resolution:- 1424 x 480

Frame rate:- 9

Length:- 00:00:16

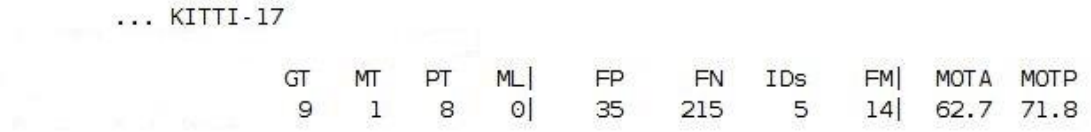




```
... KITTI-17

        GT    MT    PT    ML|     FP     FN    IDs     FM|   MOTA   MOTP
         9     1     8     0|     39    230      5     12|   59.9   71.8
```

(a)



```
... KITTI-17

        GT    MT    PT    ML|     FP     FN    IDs     FM|   MOTA   MOTP
         9     1     8     0|     35    215      5     14|   62.7   71.8
```

(b)

Figure 6.6: Tracking results from testing on KITTI17 and training on KITTI 13

(a) From proposed method (b) From previous method

Description of videos under test set:

Name- Tud crossing

Background- Stationary

Frame width, height- 826 x 571

Length of video- 00:00:22

Frame rate- 9fps

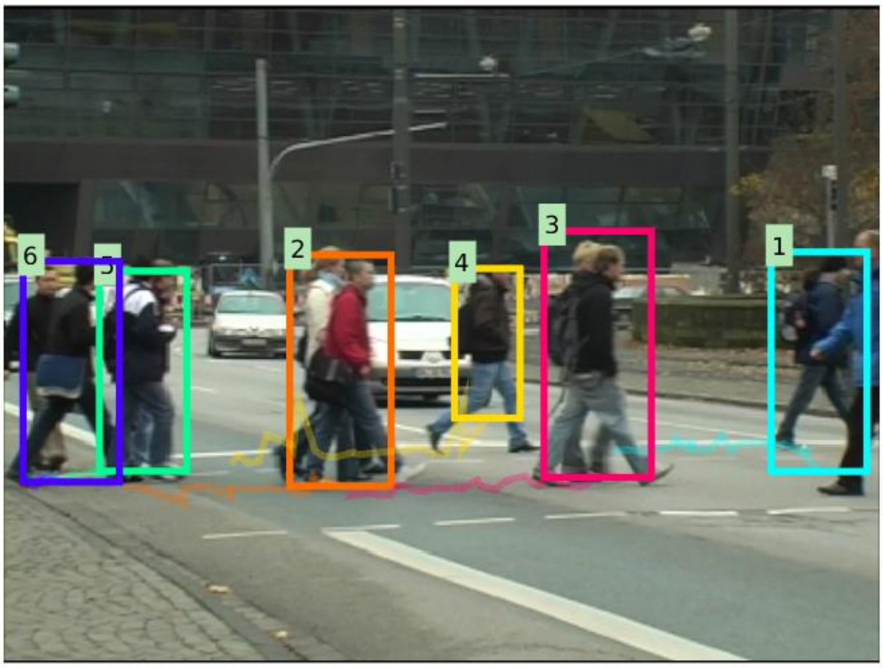


Figure 6.7: Results of tracking for tud crossing obtained when tracker trained on tud stadtmitte and tud campus.

Name- Pets09 S2L2

Background- Stationary

Frame width, height- 959 x 689

Length of video- 00:00:48

Frame rate- 9fps

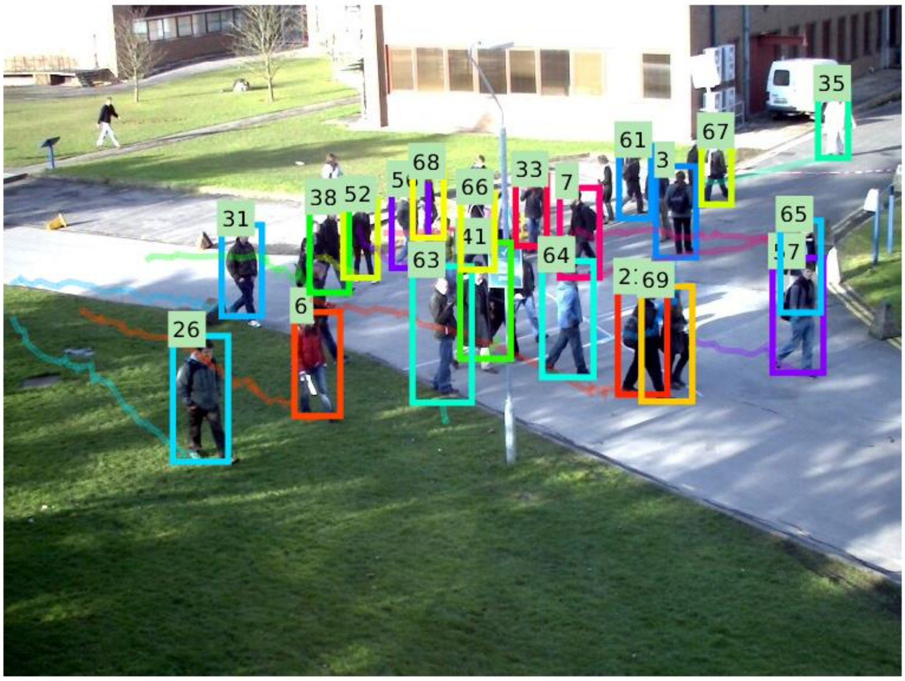


Figure 6.8: Results of tracking of pets09-s2l2 obtained when training is performed on pets09-s2l1

Name- Avg Towncentre

Background- Stationary

Frame width, height- 826 x 451

Length of video- 00:00:50

Frame rate- 9fps

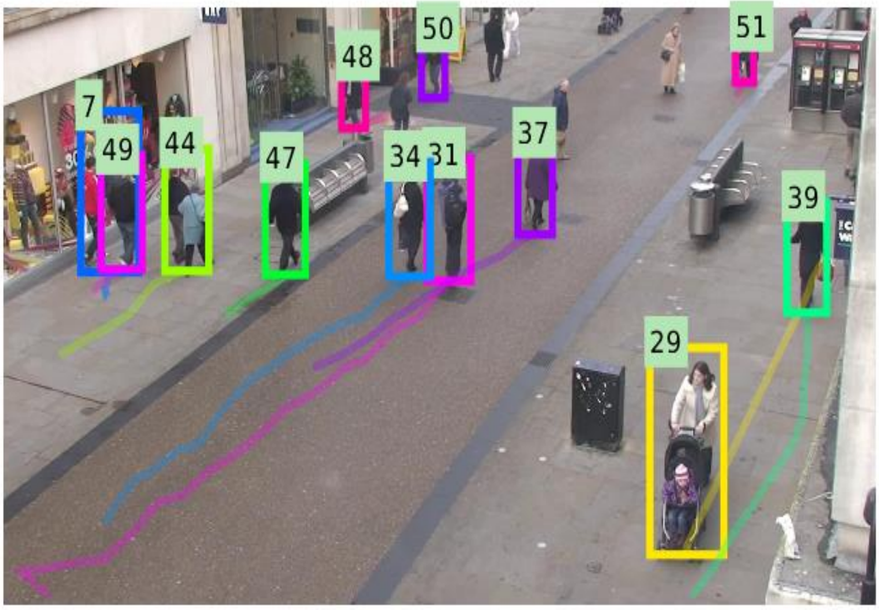


Figure 6.9: Results of tracking from sequence Avg towncentre obtained when trained on pets09-s2l1

Name- Eth Jelmoli

Background- Varying

Frame width, height- 826 x 571

Length of video- 00:00:48

Frame rate- 9fps

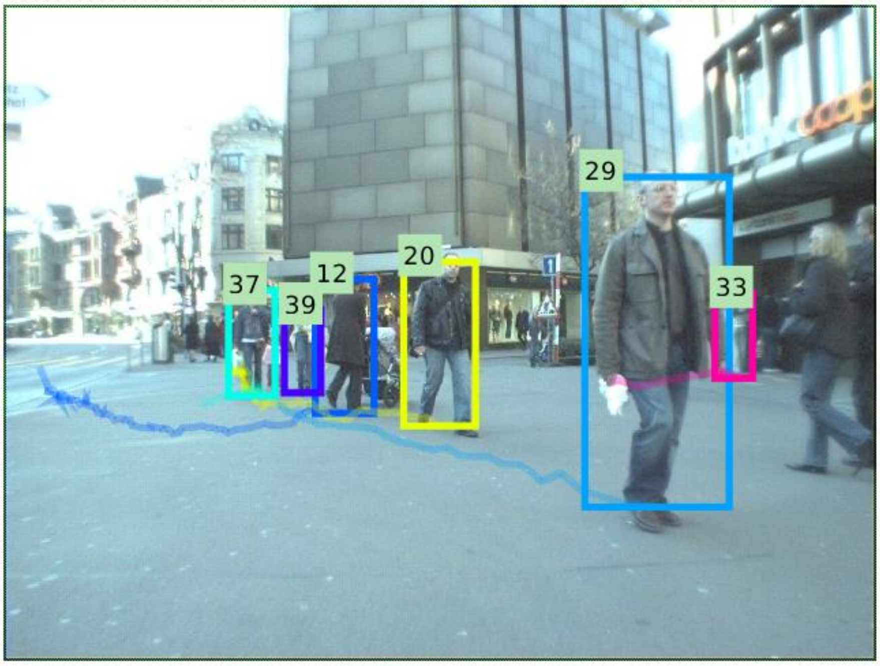


Figure 6.10: Tracking results of ETH jelmoli obtained when trained on eth bahnhof, eth sunnyday, eth pedcross2

Name- Eth Linthescher

Background- Varying

Frame width, height- 826 x 571

Length of video- 00:02:12

Frame rate- 9fps

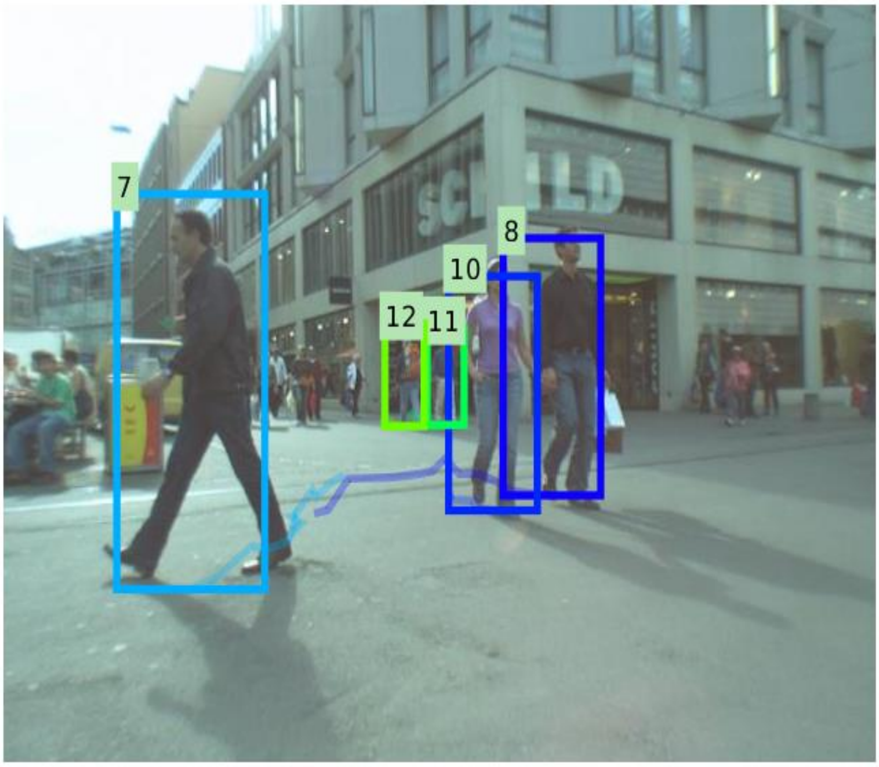


Figure 6.11: Resulting frame after tracking of Eth linthescher obtained when training is performed on eth bahnhof, eth sunnyday, eth pedcross2

Name- Eth Crossing

Background- Varying

Frame width, height- 836 x 601

Length of video- 00:02:12

Frame rate- 9fps

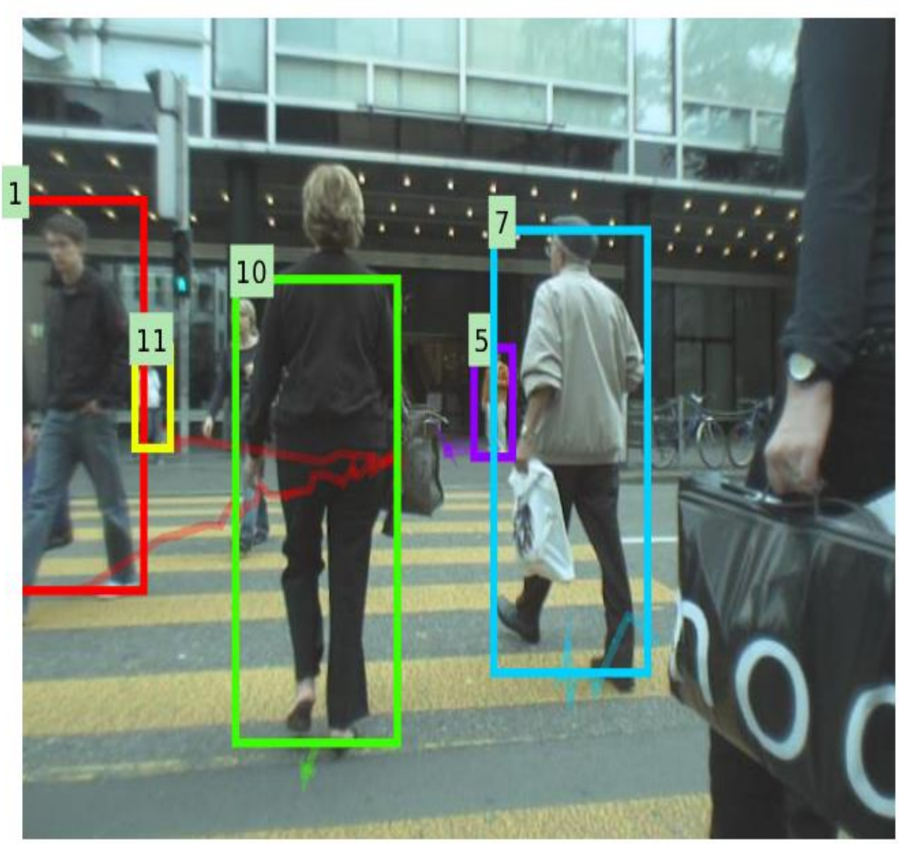


Figure 6.12: Results of tracking of Eth crossing obtained when tracker trained on eth bahnhof, eth sunnyday, eth pedcross2

Name- Adl Rundle 1

Background- Varying

Frame width, height- 826 x 451

Length of video- 00:00:55

Frame rate- 9fps

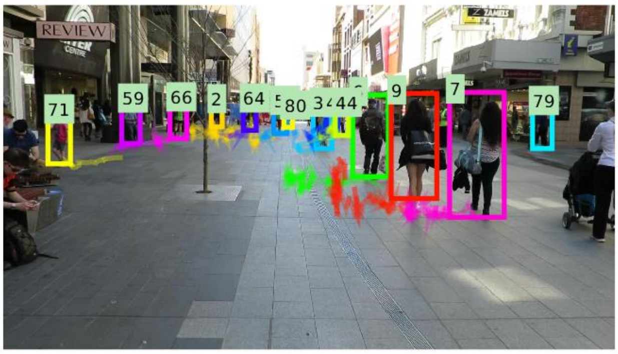


Figure 6.13: Results of Adl rundle1 obtained when trained on adl rundle6 and adl rundle8

Name- Adl Rundle 3

Background- Stationary

Frame width, height- 826 x 451

Length of video- 00:01:09

Frame rate- 9fps

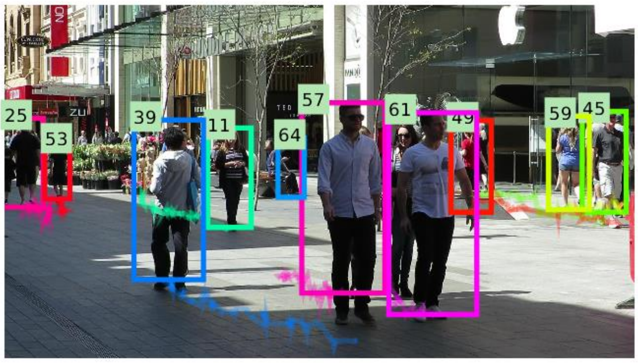


Figure 6.14: Resulting frame of Adl rundle3 obtained when training is done on adl rundle 6 and adl rundle 8

Name- Kitti 16

Background- Stationary

Frame width, height- 959 x 338

Length of video- 00:00:23

Frame rate- 9fps

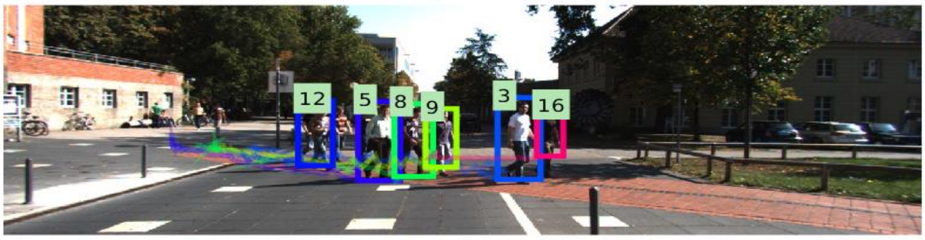


Figure 6.15: Results for kitti16 obtained when training is performed on kitti13 & kitti17

Name- Kitti 19

Background- Varying

Frame width, height- 959 x 371

Length of video- 00:01:57

Frame rate- 9fps

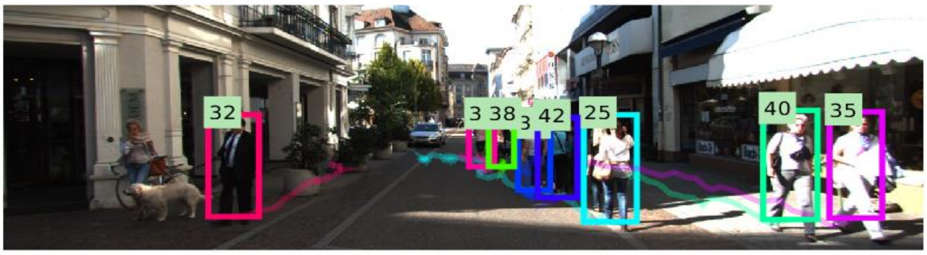


Figure 6.16: Result of kitti 19 obtained when system trained on kitti 13 and kitti 17

Name- VENICE-1

Background- Static/stationary

Frame width, height- 826 x 451

Length of video- 00:00:50

Frame rate- 9fps

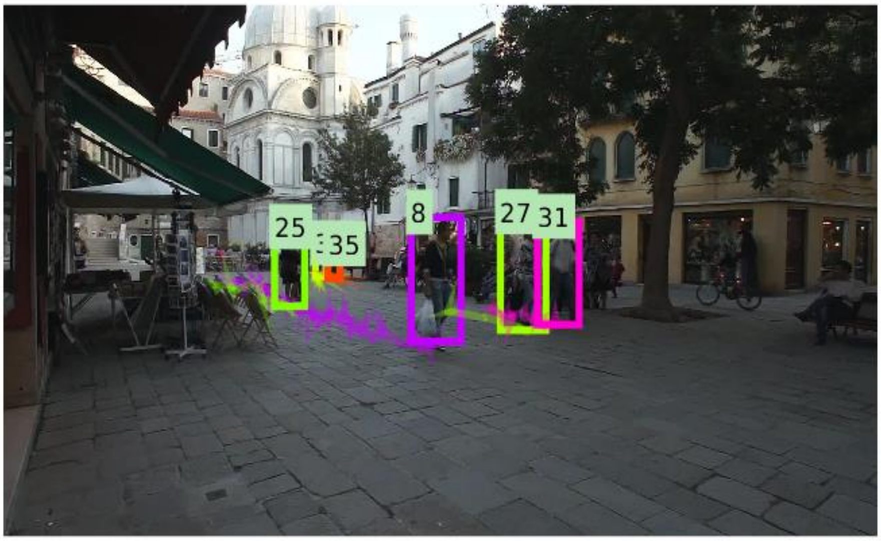


Figure 6.17: Tracking result of test video sequence venice 1 that is obtained when training of tracker is done using venice 2

CONCLUSION

In this work I have proposed a new method of object tracking by decision making using memoryless state transitions and Murty's best assignment for data association which merged with Hungarian algorithm. This system is suitable to be used in many application requiring the task of multiple object tracking. Proposed method is tested on several videos having stationary and varying background, also each video had many challenges faced by our tracker such as partial/full occlusion, change in appearance of target object and disappearance of object for some frames i.e when target object move out from camera field of view for some frames and then comes back. Under all above mentioned conditions the proposed method performs well.

FUTURE SCOPE

Further this work can be extended to make use of partially observable markov decision process where state transition depends on present and past states i.e. it has memory and makes use of history. Or can make use of semi MDP in which time taken by different actions can vary, e.g. if there are 2 different action which ultimately goes to same destination, one having +4 reward and takes 2 time steps and other having +8 reward and takes 20 time steps then the first action is preferred for execution.

This approach can be modified by use of different trackers in place of Lucas-kanade tracker. One out of several available techniques can be applied for data association part to evaluate the results. Also work can be to reduce time cost for processing each frame.

REFERENCES

[1]. MOT benchmark dataset available on https://motchallenge.net

[2]. James and D. Ramanan. “Tracking as online decision making: learning a policy from streaming videos with reinforcement learning”, International conference on computer vision, in 2017.

[3]. Zdenek, Kalal, K. Mikolajczyk, and J. Matas. “Tracking-Learning-Detection” IEEE transaction on Pattern Analysis and Machine Intelligence, 2012.

[4]. S. Avidan “Support vector tracking”. IEEE transactions on Pattern Analysis and Machine Intelligence, in 2004.

[5]. Keni and Rainer. “Evaluating multiple object tracking performance: the clear mot metrics”. EURASIP Journal on video & image Processing, 2008.

[6]. R.Collins.“Multitarget data association with higher-order motion models”,in Computer Vision and Pattern Recognition,2012.

[7]. P.Dollar, Christian, B.Schiele, and Perona.“Pedestrian detection: An evaluation of the state of the art”.IEEE Transactions on PAMI, 2012.

[8]. S. Gu, Y. Zheng, and C. Tomasi “Efficient visual object tracking with online nearest neighbor classifier”, ACCV 2011.

[9]. Harold Kuhn“The hungarian method for the assignment problem.” Naval research logistics quarterly 1955.

[10]. Laura Leal-Taix´e, Anton Milan, Ian Reid, Stefan Roth, and Konrad Schindler. “MOT challenge 2015: Towards a benchmark for multi-target tracking” arXiv:1504.01942, 2015.

[11]. L. Matthews, T. Ishikawa, and Baker “The template update problem. Pattern Analysis and Machine Intelligence”, IEEE Transactions 2004.

[12]. James Munkres.“Algorithms for the assignment and transportation problems.” Journal of the society for industrial and applied mathematics 1957.

[13]. J. Supancic, Peter Carr and D. Ramanan.“Why can’t we all just get along? effective identity-aware multi-object tracking from sensored annotation?” in International Conference on Computer Vision, 2017.

[14]. Yi Wu, J. Lim, and Ming-Hsuan Yang.“Online object tracking: A benchmark”. Computer Vision and Pattern Recognition, 2013.

[15]. Y.Xiang, Alexandre Alahi, and S. Savarese.“Learning to track: Online multiobject tracking by decision making.” ICCV, 2015.

[16]. A. Yilmaz, O. Javed and M. Shah. “Object tracking: A survey” Acm Computing Surveys 2006.

[17]. V. Tarasenko and Dong won park. “Detection and tracking over image pyramids using Lucas and Kanade algorithm” International Journal of Applied Engineering Research, 2016.

[18]. Simon Baker, Iain Matthews, Lucas-Kanade“20 Years On: A Unifying Framework.” Carnegie Mellon University Robotics Institute 2002.

[19]. Jean-Yves Bouguet “Pyramidal Implementation of the Lucas Kanade Feature Tracker. Description of the algorithm”. Intel Corporation (2001).

[20]. S. David, E. Kushilevitz and Y. Mansour. “Online learning vs offline learning”, Machine learning,1997.

[21]. Z. Kalal, K. Mikolajczyk and J. Matas. “Forward-Backward Error: Automatic Detection of Tracking Failures” Pattern Recognition 20th International Conference, 2010.

[22]. I.J. Cox and M.L. Miller.“On finding ranked assignments with application to multi target tracking” IEEE transaction AES, 1995.

[23]. R. Bellman, “A markovian decision process.” Journal of Mathematics and Mechanics, 1957.