



UNIVERSITY OF BELGRADE
FACULTY OF ELECTRICAL ENGINEERING



Abdalgail Alsagair Mohamed Abdulla

One Particle Filter Based Algorithm of Tracking of a Moving Object in the Sequence of Images

Doctoral Dissertation

Belgrade, 2022

УНИВЕРЗИТЕТ У БЕОГРАДУ
ЕЛЕКТРОТЕХНИЧКИ ФАКУЛТЕТ

Абдалгалил Алсагаир Мохамед Абдула

Алгоритам праћења покретних објеката у
секвенци слика применом честичног филтра

Докторска дисертација

Београд, 2022

Предлог састава Комисије за преглед и оцену докторске дисертације (предлаже Ментор):

1. Др Бранко Ковачевић, професор емеритус,
(Универзитет у Београду, Електротехнички факултет, ментор)
Dr. Branko Kovačević, professor emeritus,
(University of Belgrade, School of Electrical Engineering, mentor)

2. Др Жељко Ђуровић, редовни професор,
(члан комисије са изабраног модула кандидата - препорука, није обавезно)
Dr. Željko Đurović, professor,
(University of Belgrade, School of Electrical Engineering)
(Committee member)

3. Др Зоран Бањац, Научни сарадник, Институт Влатаком, Београд
(члан комисије који није у радном односу на ЕТФ-у)
Dr. Zoran Banjac, research associate,
(Vlatakom Institute, Belgrade)
(Non-ETF board member)

4. Др Милан Прокин, редовни професор,
(члан комисије са изабраног модула кандидата - препорука, није обавезно)
Dr. Milan Prokin, professor,
(University of Belgrade, School of Electrical Engineering)
(Committee member)

5. Др Влеко Папић, ванредни професор,
(члан комисије са изабраног модула кандидата - препорука, није обавезно)
Dr. Veljko Papić, associate professor,
(University of Belgrade, School of Electrical Engineering)
(Committee member)

Title of doctoral dissertation:

One Particle Filter Based Algorithm of Tracking of a Moving Object in the Sequence of Images

Abstract:

There are a continuously increasing number of applications where moving objects should be tracked in a video sequence. This research work has been directed to and focused on video sequence monitoring and following up of automobiles in artificial and real-world traffic scenarios. Bearing in mind the rapidly growing scope of applications in the areas of automatic monitoring of traffic situations on the roads, as well as on the control of autonomous vehicles in traffic, the attention was focused on the problem of moving vehicle video tracking as the starting phase of many of the tasks of this type. The concept of “Particle Filter” has been adopted as the basis for our approach in the estimation of an object's position in an image, because of its verified abilities in solving the tracking problems in complex situations regarding motion dynamics, as well as from the aspect of disturbance nature. As in the cases already existing in the literature, we have based our approach on the choice of appropriate image features that should be used as the object's appearance model. Algorithms supported by Particle Filter (PF) are more and more frequently used for these functions. The suggested tracking algorithm here is based on the implementation of PF using three independent features of the image at the same time. (color contents/histogram, objects' contour extracted by the usage of the image gradient, and texture defined by Grey Level Co-occurrence Matrix). All three features are represented by appropriate histograms that are associated with the contents of a rectangular window that contains both the vehicle that is tracked and the surrounding area.

Based on individual estimates produced by every single feature, the final estimate is made by their weighted average. It's been adopted the mechanism of adaptation of weighting coefficients for particular image features, including the cases that some of them should be excluded from the calculation if their level of confidence is below some specified threshold. The quality of a specific feature is determined by calculating average similarities between the reference window and the collection of windows on particles' locations, and according to that, the weighting coefficients modify adaptively. The tracking accuracies of single-feature and three-features based filters were tested utilizing a variety of artificial and real-world traffic sequence scenarios that enclose different common perturbations (shadows, variable background, partial and full occlusions, maneuvering, etc.).

Using the three-feature-based filter demonstrated the improvement of tracking accuracy in comparison to any of single-feature based ones using video sequences illustrating typical traffic scenarios.

Keywords: Object tracking, particle filter, image features, color histogram, texture, Grey Level Co-occurrence Matrix, image gradient, histogram of oriented gradients.

Scientific area: Computer vision

Narrow scientific area: Moving object monitoring and tracking

Наслов докторске дисертације:

Алгоритам праћења покретних објеката у секвенци слика применом честичног филтра

Апстракт:

У савременом свету континуално расте број примена у којима треба пратити покретне објекте у видео секвенци. Овај истраживачки рад је усмерен на праћење аутомобила у видео секвенцама које су рачунарски синтетизоване или потичу из реалних саобраћајних сценарија. Имајући у виду све већи обим примене у областима аутоматског надзора саобраћајне ситуације на путевима, као и управљања аутономним возилима у саобраћају, пажња је усмерена на проблем видео праћења покретних возила, као на почетну фазу многих задатака овог типа. Концепт „честичног филтра” (англ. “Particle Filter”) усвојен је као основа за приступ у процени положаја објекта на слици, због његових проверених способности у решавању проблема праћења у сложеним ситуацијама у погледу динамике кретања, као и са аспекта природе поремећаја. Као и у већ постојећим случајевима у литератури, предложени приступ се заснова на избору одговарајућих карактеристика слике које треба користити као модел репрезентације објекта. Алгоритми на бази честичног филтра (ПФ) се све чешће користе за ове намене. Овде предложени алгоритам праћења је заснован на имплементацији ПФ-а истовремено користећи три независне карактеристике слике (садржај боје/хистограм, контуре објеката издвојене употребом градијента слике и текстуру описану матрицом узајамног појављивања нивоа сивог). Све три карактеристике су једнообразно представљене одговарајућим хистограмима који се односе на садржај правоугаоног прозора који обухвата праћено возило и његову локалну околину.

На основу засебних процена које производи свака појединачна карактеристика, коначна процена се прави на основу њиховог пондерисаног усредњавања. Усвојен је механизам прилагођавања тежинских коефицијената за одређене карактеристике слике, укључујући случајеве да неке од њих треба потпуно искључити из прорачуна ако је њихов ниво поверења испод одређеног прага. Квалитет специфичног обележја се утврђује израчунавањем просечних сличности између референтног прозора и скупа прозора на локацијама честица, па се према томе адаптивно мењају тежински коефицијенти при усредњавању. Тачност праћења филтера на бази појединачне карактеристике и на бази симултаног коришћења све три, тестирана је коришћењем различитих саобраћајних сценарија (синтетичке и реалне секвенце) које обухватају различите типичне поремећаје (сенке, променљива позадина, делимичне и потпуна заклоњеност, маневрисање, итд.).

Примена филтера заснованог на истовременом коришћењу три карактеристике слике показала је побољшање тачности праћења у поређењу са било којим од филтера заснованих на појединачној особини, на свим анализираним видео секвенцама које илуструју типичне саобраћајне сценарије.

Кључне речи: Праћење објеката, честични филтер, карактеристике слике, хистограм боја, текстура, матрица ко-појављивања нивоа сивог, градијент слике, хистограм оријентисаних градијената.

Научно област: Рачунарска визија

Уже научно област: Надзор и праћење покретних објеката

Statements of gratitude,

I am extremely grateful to my mentors and supervisors, Prof. Dr. Branko Kovacevic (full professor) and Prof. Dr. Stevica Graovac (associate professor). I take this opportunity to thank them so much for faithful and professional assistance, continuous support, valuable suggestions which have directed my research to the successful completion of this doctoral dissertation, and patience during all the period of my study especially during the preparation of the dissertation.

I would like to express my full thanks to Dr. Veljko Papic (associate professor) for his professional help and extremely important comments and suggestions during the preparation of the journal paper and this doctoral dissertation.

I would also like to extend my sincere thanks to Dr. Alexandra Marianovic (assistant professor) for her continuous and unlimited support to me in solving the programming problems that I faced during the doctoral study period.

I also want to thank the other professors for their assistance in the completing the thesis and the knowledge gained during the doctoral study.

I also wish to express my profound gratitude to my friends and to all those who have kindly assisted me in this work.

My appreciation also goes out to the ministry of higher education, the State of Libya for their trust by presenting the financial support of my Ph.D. scholarship.

Finally, I owe a great deal of gratitude to my wife Aisha and daughter Tasnim for their understanding during the different stages of studies and writing this doctoral dissertation.

Table of contentes

1 Introduction	2
1.1 Concept of visual object tracking (VOT) in computer vision	2
1.2 Visual object tracking's applications	4
1.3 Difficulties and challenges	5
1.4 Contributions	8
1.5 The structure of this thesis	8
2 Tracking of Object in Video Images Sequence	10
2.1 Principle of Visual Object Tracking and Related work	10
2.1.1 Object detection	11
2.1.2 Object modeling and representation	15
2.1.3 Object tracking	19
2.2 Tracking strategies	19
2.2.1 Model-based tracking	20
2.2.2 Region-based tracking	20
2.2.3 Contour based tracking strategy	21
2.2.4 Feature-based tracking	22
2.3 Appearance model in object tracking	23
2.4 Motion model	25
2.5 Disturbances	25
2.6 Choice of a Particle Filter (PF)	26
2.7 Fusion of multiple cues	26
3 Tracking of Moving Object Based on Multiple Image Features Using Particle Filter	28
3.1 Particle Filter Applied in Video Object Tracking	28
3.1.1 Bayesian's filter	29
3.1.2 Particle filtering	30
3.1.3 Particle filter algorithm	32
3.2 State Transition	34
3.3 Visual Tracking Based on Image Features	35
3.3.1 Color feature	35
3.3.2 Edge (contour/border) feature	38
3.3.3 Texture feature	39

3.4 State estimation	42
3.5 Resampling	44
3.6 Similarity measure	45
3.7 Multi-Feature-Based Tracking Algorithm	46
4 The performance evaluation of the proposed SFPP algorithm	54
4.1 Testing the environment for object tracking using different features based Particle Filter	54
4.2 Testing the proposed tracking algorithm SFPP using multiple descriptors of the image independently	55
4.2.1 Synthetic Traffic scenarios generation for testing of the tracking algorithm	55
4.2.2 Single object tracking in real-video sequences using three independent image descriptors	60
5 Performance Evaluation of MFPP Tracking Algorithm in Comparison to SFPP Algorithm	67
5.1 Impacts of varying some algorithm's parameters	68
5.1.1 Impact on artificial video sequences	68
5.1.2 Impact on real video sequences	74
5.2 Experimental Comparison of MFPP and Different SFPP Algorithms	81
5.2.1 Synthetic video sequences	81
5.2.2 Real video sequences	87
6 Conclusion	98
7 Bibliography	100

Chapter 1

Introduction

1 Introduction

1.1 Concept of visual object tracking (VOT) in computer vision

Visual object tracking is one of the most substantial missions in computer vision, aiming at the estimation of kinematic parameters (position, velocity, the direction of motion, etc.) and other relevant information, that is describing the moving object in a set of image sequences captured by a camera [1]. Object tracking in computer vision is an approach of automatic determining or estimating an object's trajectory (location) within subsequent video image sequences based on previous scene information. Therefore, the general objective is to segment the specified object from the scene and keep track of it or follow its states from frame to frame over time, to extract useful information and recognize the object's behavior using some algorithms. Objects in the scene might be moving vehicles on the ground, air, sea, walking people, or animals moving in the natural environment, etc. [2], [3], [4].

Tracking moving objects in the domain of computer vision is the required and essential part of many applications, such as object tracking, traffic monitoring systems, video surveillance systems, sports reporting, security-related tasks, medical imaging, and so on. Most of these tasks are based on the processing and analysis of video image sequences, which is the source of information [5]. In this context, vehicle tracking serves as the foundation for higher-level missions such as traffic control, event detection, information and data extraction from traffic flow, etc. [6], [7], [8]. Typical problems for reliable vehicle tracking in these circumstances are variable background and shadows, variable existence of other close objects (moving or stationary), partial or even full occlusions by elements of the scene, variable size of an object during the sequence, etc. [9], [10], [11]. Moreover, visual tracking is usually affected by the problems of camera vibration caused by the wind, large lighting changes, and other similar technical factors.

Tracking methods are usually categorized as generative and discriminative ones [12]. Generative tracking approaches typically are used to search for the image region with minimal reconstruction error, whilst discriminative tracking is dependent on the separation of the target from the background. One generative method is adopted, assuming that the window (sub-image) of interest consists of a tracked vehicle and surrounding local background. Features characterizing this window form the mathematical representation of it.

This thesis proposes a novel method and algorithm for tracking a moving vehicle on the road using the video images captured by a video camera. The objective is to create a detecting and tracking approach for vehicles utilizing multiple features. The tracking approach is generally divided into four parts:

- Target initialization (specifying the initial state);
- Appearance modeling (extracting the visual features);
- Prediction (estimation of target motion), and
- Updating (determining the target position).

An ideal object tracking algorithm should satisfy the following requirements [13]:

- Detection of the tracked object in the first frame of image sequences should be done (manually or automatically),
- It must work in the presence of various disturbances such as background clutters, shadows, scale changes, or when the tracked object is hidden or departs outside the video frame borders, etc.
- After the termination of occlusion, it should be able to pick up the object that has been lost during occlusion or in-between frames,
- Work in real-time – according to the dynamics of target motion in the scene.

Object tracking is a hard mission and challenging for any visual tracking algorithm in the existence of many disturbances, but many ways can be used to improve the object tracker. In this thesis, a novel algorithm is created to overcome the problems of background clutter, illumination change, or occlusion. This algorithm was used to track objects in the created synthetic image sequence as well as in the sequences of real video frames.

Regarding the chosen tracking algorithm concept, we have been motivated by the fact that a Particle Filter (PF) is extremely robust for nonlinear and non-Gaussian dynamic state estimation and works well where clutter, noisiness, and occlusions are present.

Using the PF approach, physical system dynamics could be modeled rightly if the elements of non-linearity and non-Gaussian type disturbances are included. The work presented in [14] is usually referenced as an origin in this particular area. Particle Filters are consecutive Monte Carlo methods that rely on point mass representations of probability densities that could be used in any state model [11].

In the status of tracking a moving target, the state model includes the target's location (coordinates of the image) and the rate of its change (velocity), while the size of the object also could be added as an additional part of the model. In the context of the processing of a series of images, probability density functions consist of histograms representing the distributions of particular image features [15].

The very first image feature used for these purposes was the histogram of colors (in RGB or HSV space). Though color-based trackers are robust in cases of partial occlusion, object deformations, rotation, and size invariance, the practice has shown that PF based solely on this feature is ineffective in many cases, where the tracked window's content is subjected to changes in some conditions, such as strong shadows, luminance variations, or if the background and the tracked object have similar colors [9]. Other features such as gradients and texture are used in some approaches. All of these features have shown their sensitivity and none could be suggested as the universal candidate. Gradient-based cues are sensitive to the existence of other “non-target” objects within the window, while their calculations might require an enormous computational effort. The texture-based cues are relatively less susceptible to changes in illumination, but they are extremely sensitive to cluttered backgrounds [16].

The reasonable next step is the merging of multiple cues to adjust to their sensitivities. There are many approaches of this type, mostly combining the color feature with some representation of the object's edge [9], [17], [18] or, color and texture [5], [19]. Because the PF algorithm is commonly a time-consuming algorithm, one should define appropriately and properly the parameters of the integrated algorithm, in addition to finding the compromise and fulfillment of a balance between the number of calculations (which increases as new features are added) and the amount of time it takes,

and therefore achieving the tracking accuracy. In the algorithm shown here, the fusion is made by weighted averaging of estimates, supported by the usage of the three features (color, texture, and edge descriptors). Using a relatively small number of particles, the results obtained by this “Multi-Feature Particle Filter” (MFPF) have shown advantages compared with all separate “Single-Feature Particle Filter” (SFPF) algorithms and were still acceptable from the total computing time aspect of view.

1.2 Visual object tracking’s applications

Visual object tracking is a significant sub-domain in computer vision. It has many applications that have been used with fixed cameras or movable ones, and it has been seen as an essential aspect of security and surveillance systems.

Nowadays, visual tracking has used in an enormous range of fields, like industrial production, military establishments, planning in civil engineering, multimedia, etc. Some of its important applications are given below and shown in Fig.1.1:

- (1) Automated video surveillance and security: It is used to find out, recognize and track objects in a succession of images. It is designed for security and monitoring the movements in an area (streets, crowded gathering places, railway stations, car parks, shopping centers, etc.), specifying the moving objects and recording any suspicious event. The system needs to differentiate between existing things that request a good visual tracking system [2].
- (2) Traffic monitoring: it could be described as a means used to managing, monitoring traffic, security, and safety on roads or highways using cameras. As an example, detection of traffic accidents, monitoring of traffic flow, or if a vehicle breaks the traffic laws; so it could be tracked easily if the street is supplied with a surveillance system that is supported by a tracking system [2].
- (3) Robotic vision: It could be used in the guidance and control system of robots (industrial and humanoid). The guidance system of the robot requires identifying different obstacles in its route to avoid a collision. If the obstacles are movable objects, then it requires a visual tracking system [2], [20].
- (4) Vehicle tracking: Visual object tracking could be applied to vehicle tracking using a fixed camera or movable one, such as tracking a vehicle on a road by a fixed camera beside the road or on the bridge or an unmanned aerial vehicle [20].
- (5) Medical diagnostic systems: Computer vision has derived its importance in the medical area for diagnosis of different diseases, while observing the motion of naturally existing or artificially inserted particles/liquids. It can help doctors find some diseases, which result in fast therapy.
- (6) Military institutions or government: There are widespread government and military applications which face very different security issues. The security and protection of government and military sites may include a variety of tasks in various areas, such as

monitoring the surrounding environment and prohibited location entry such as military bases, airports, safe transportation processes, seaports, dams, bridges, tanker loading terminals, terrorist hazard prevention, and so on [2].

- (7) Traditional ground-to-air and air-to-ground missile homing application are used in order to destroy the enemy's objects.
- (8) Video games: Visual object tracking algorithm could be expanded to animation and video games [2], [20].
- (9) Activity recognition: Visual object tracking might be used for activity recognition, e.g. indoor and outdoor monitoring, such as human activity recognition, e.g., monitoring human presence and traffic in and around buildings [20]. A very exemplary application is the monitoring of different parts of a commercial store for potential shoplifters.

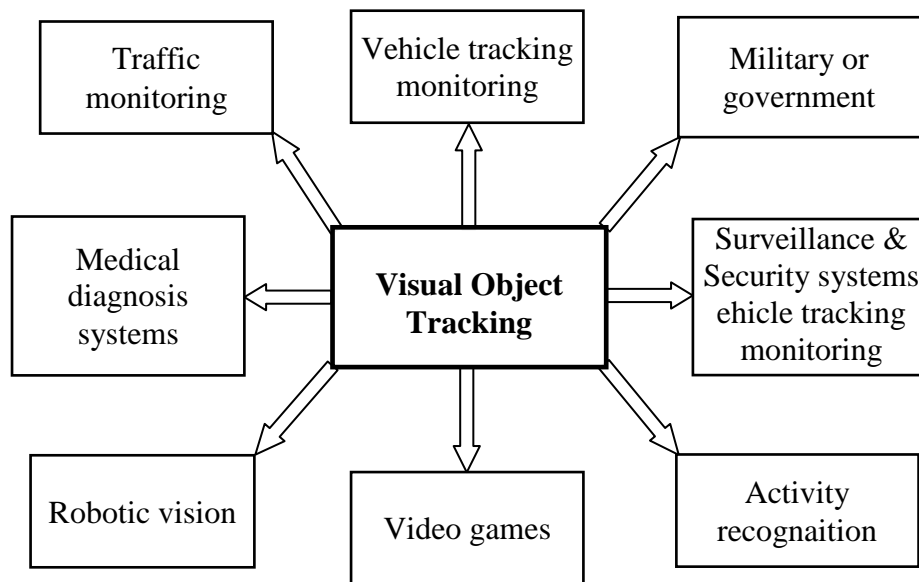


Fig.1.1 Some visual object tracking applications

1.3 Difficulties and challenges

Visual object tracking in video sequences captured by cameras faces various difficulties and challenges. Some of these problems that occur in the object tracking process are discussed below and shown in Fig.1.2.

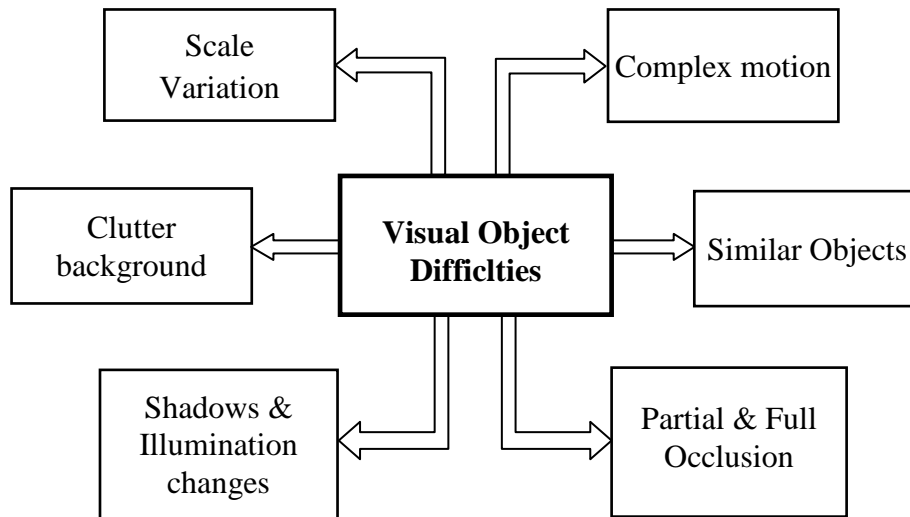


Fig.1.2 Some visual object tracking difficulties

- **Shadows and Illumination variation:** the shadows produced by objects on or beside the road (buildings, trees, clouds, etc.), make the task of detecting and tracking objects more difficult. Also, one of the major challenges is that changes in outdoor lighting conditions for any reason can have an effect on the object's appearance and make it difficult to discriminate it because numerous features of the target that are noticeable in high light become ambiguous in lower illumination. Fig.1.3 shows a typical example of a shadow challenge.



Fig.1.3 An example of shadow and illumination [21]

- **Complex motion and drifting problem:** When the target abruptly changes its speed and direction, such as when a car is changing its motion or when people are slipping in the snow. In that case, the tracking process becomes a more difficult mission due to the inaccurate approximation of the movement type [20], [22].
- **Occlusion:** in situations where the moving target vanishes and reappears in the scene, or is occluded partially or fully by another object for a while before reappearing at a different location, the tracking issue turns more difficult [23]. In partial occlusion, a few or several parts of the object might be hidden by other objects, and the target could be tracked, while in the case of complete occlusion, the object would completely disappear, which leads to difficulties

for the object tracking, since there is no particular technique to overcome it. As a consequence, the strategies are selected based on the kind of target, conditions, and surrounding environment. Common examples of complete occlusions in traffic are vehicles occluded by bridges, other vehicles, or trees beside the road, all of which are typical in traffic monitoring [24]. Fig.1.4 shows the partial and full occlusions as an example.



Fig.1.4 An example of partial and full occlusion illustration

- **Scale variation:** when an object moves toward the camera, its size increases, while if it moves away from the camera, its size decreases, particularly for a fixed camera. Therefore, the change in the target's size in the image should be addressed by the tracking strategies [2], [20]. Fig.1.5 displays an example of the size decreasing of the moving vehicle.



Fig.1.5 An example of scale variation

- **Background clutter:** if the background of the scene contains many transient other objects, particularly in natural outdoor environments, it is referred to as a cluttered background, or dynamic background, such as the motion of the leaves of the trees in the wind, swaying trees, humans or animals crossing the road, traffic lights, or waves of water, and so on. Sometimes it is easy to handle these problems, but sometimes it is difficult to deal with them.
- **Identical objects:** if two similar objects have moved close to each other, then the tracker of the target can leap to the other object. For this reason, the tracked object might be lost and track the wrong one [22]. This problem is known as the hijacking problem.

As a result, visual object tracking suffers from a lack of robustness due to these disturbances. From a mathematical standpoint, most of the visual tracking models encountered can be considered as non-Gaussian, non-linear, multi-modal models, or some combination of them [17].

In the literature, there are numerous algorithms that have been evolved and performed to cope with difficulties that appear in the video tracking process, such as distance and orientation gradient histogram, mean shift, Scale Invariant Feature Transform (SIFT), Random Sample Consensus, optical flow, etc. [2].

1.4 Contributions

Although, in the literature, many tracking algorithms use one feature or two fused features which work in some environments, the aim in this work is to create a robust algorithm for detecting and tracking vehicles in a video surveillance environment using more than two features, and taking into consideration some different vehicle tracking problems mentioned in previous sections.

For this purpose, the first contribution of this work was the generating of a video sequence of different artificial traffic scenarios on a two-lane road using MatLab. Five scenarios have been proposed and provided for testing the invented tracking algorithm and indicated as cases for particular traffic motions in different conditions, which have been demonstrated in chapter 4.

The *second* contribution is to design an algorithm for vehicle tracking in a generated synthetic video sequence using a particle filter. This algorithm can be applied to a real-world video sequence as well, which is presented in the results section later.

The proposed tracking algorithm is based on simultaneous usage of three different image features – color, edge orientation, and texture, and fusing these three cues to deal with the disadvantages of individual features, and to compensate their sensitivities produced by the circumstances of the surrounding environment. In the proposed algorithm, the fusion of the three descriptors is made based on the weighted average of estimates of the three features (color, texture, and edge).

1.5 The structure of this thesis

The rest of this thesis has been organized as follows: In chapter one, the concept of visual object tracking, applications, and related problems are explained. Chapter two contains an overview of relevant topics in various sub-components in the literature relevant to object detection and tracking in a video series, which supports the contributions in subsequent chapters. The fundamental principles of particle filter-based object tracking, as well as the selection of particular image's features, their formalization, and features' individual efficiency, have been presented and discussed in chapter three. In chapter four, the new proposed algorithm that consists of integrating three individual SFPP using descriptors' weights adaptation is explained and the flowcharts of SFPP and MFPP algorithms are included as well. In chapter five, typical traffic situations were presented as the verification cases as well as the set of the obtained results as the outcome of features incorporation and to illustrate the performance of our proposed algorithm. And furnish the appropriate discussion. Finally, chapter six provides conclusions drawn from this thesis, future research, and work. A list of the references is given in the last part of the thesis.

Chapter 2

Tracking of Object in Video Images Sequence

2 Tracking of Object in Video Images Sequence

2.1 Principle of Visual Object Tracking and Related work

Visual object tracking has been developed through the past decades, and many different methods have been suggested in a huge number of references in the literature related to moving object detection and tracking in video sequences [25], [26]. These methods mainly vary from one to another about how they address the following issues:

- How to exemplify the tracked object?
- Which merits of the image should be utilized?
- How should the object's form, appearance, and motion be represented in the model?

The answers to these questions generally rely on the condition and surrounding environment in which the tracking process is accomplished [27]. Some tracking models have been evolved which try to answer these questions for various scenarios, such as Markov chain Monte Carlo, optical flow, Kalman filter, particle filter, which are all well-known image processing methods for object tracking purposes [26].

Vehicle tracking in real traffic flow with dynamic changes could be considered as a non-linear case. Even though the extended Kalman filter is able to cope with the non-linear state model, it is still facing the problem of diverging when the approximation of non-linearity is made in an inaccurate manner. Thus, for this study, the particle filter (PF) was selected as a trustworthy tracking algorithm for vehicle tracking due to its robustness and its ability to deal with non-linear and non-Gaussian problems [26].

The basic stages for object tracking generally consist of three steps: (1) object detection, (2) object representation, and (3) object tracking [28], [29], [30] as shown in Fig. 2.1

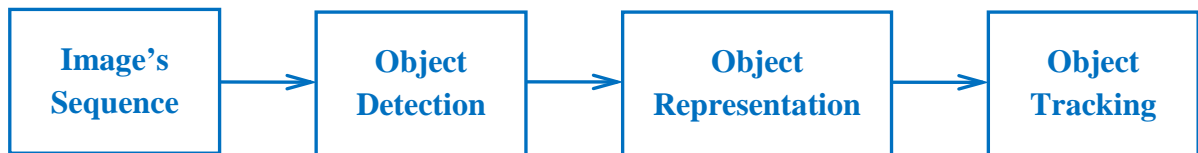


Fig. 2. 1 *The essential phases concerned in the object tracking*

2.1.1 Object detection

Object detection is the first step in identifying the tracked object in a video series, and it's the process of determining the location and distinguishing the associated pixels necessary for the objects of concern in the frames. It is an essential condition for the tracking process procedure [31], [32]. In general, there are two techniques for object detection mostly applied to commencing the tracking process:

- (1) Manually detecting and bounding the object we want to track by a rectangle and determining its location in the first frame, then allow the tracking algorithm to detect the proposed features to follow the object in the subsequent frame and
- (2) The object is automatically detected using proposed features such as color, texture or edge, etc. [31].

There are numerous different approaches that are applied to detecting moving objects, which have been developed and mentioned in the literature. Some of the main and leading techniques in discovering moveable objects are [28], [29], [31]:

- Background subtraction,
- Frame difference,
- Temporal differencing,
- Optical Flow,
- Kalman filter,
- Particle filter, and
- Others.

- i) **Background Subtraction:** In video sequences with a static background, background subtraction is extremely used [31]. It segments the image into two parts: foreground and background. The foreground consists of moving objects like moving vehicles, people, while the background includes static objects, such as trees, buildings beside the road, or obstacles like the stopped cars, etc.

In this technique, the reference background image is captured first without the existing of the tracked object in the scene. The moving objects or foreground objects are revealed by subtraction between the current image frame and the reference background image frame.

In order to discover and identify the moving objects, a threshold value must be specified. The threshold is used to determine whether pixels in an image belong to the foreground or the background.

The pixels at intensity levels above the threshold are labeled as foreground, while the pixels of lower intensity level than the threshold are labeled as background. As in practice, by time elapsing, the background of any scene progressively changes, and to bypass incorrect detection of objects, the reference background image should be updated from time to time. Background subtraction is simple and straightforward to implement [33] and performs well against a static background and with several moving objects, but the drawback is time-consuming, and their sensitivity to the changes of the illumination and external environment [30], [32], [34]. An

illustrative example of this principle is given in Fig. 2.2. Two moving cars on the road (Fig. 2.2.b) could be discriminated by simple and straightforward subtraction of the static scene (Fig. 2.2.a), The image obtained from the background subtraction approach (Fig. 2.2.c) obviously shows the objects of regard.

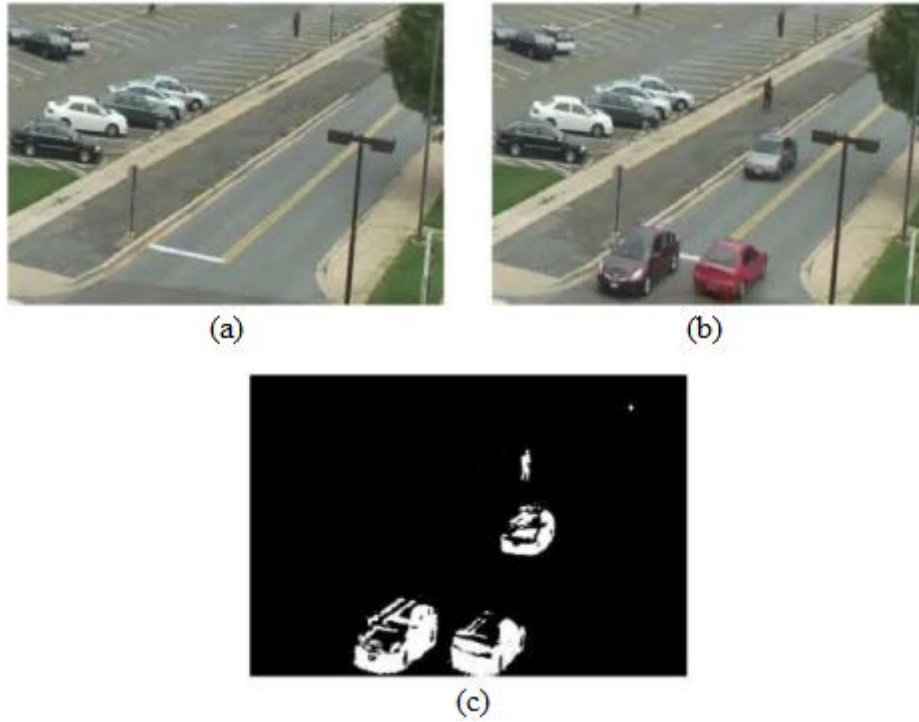


Fig. 2.2 Background subtraction example, (a) The background scheme (b) The new frame (c) The background subtraction results [34]

- ii) **Frame difference:** The frame difference approach is also a subtractive method, like background subtraction. This approach is mainly carried out by N consecutive frames, in which usually the comparisons are done by computing the differences between two or three consecutive frames pixel-by-pixel as well as threshold assignment, leading to revealing the objects and movement in the scenario [28], [29], [30]. Mathematically, the difference between the current and previous frames is calculated by Equations (2.1) and (2.2).

$$\text{Foreground pixel: } \left| F_c - F_p \right| \geq \text{Threshold} \quad (2.1)$$

$$\text{Background pixel: } \left| F_c - F_p \right| < \text{Threshold} \quad (2.2)$$

Where F_c and F_p are current and previous frames respectively.

This method is straightforward to implement and can proceed in real-time with a lower delay of only one frame for a simple version [35].

Fig.2.3.a and Fig.2.3.b show a person in two successive frames, and the result of applying the frame difference approach is illustrated in Fig.2.3.c.

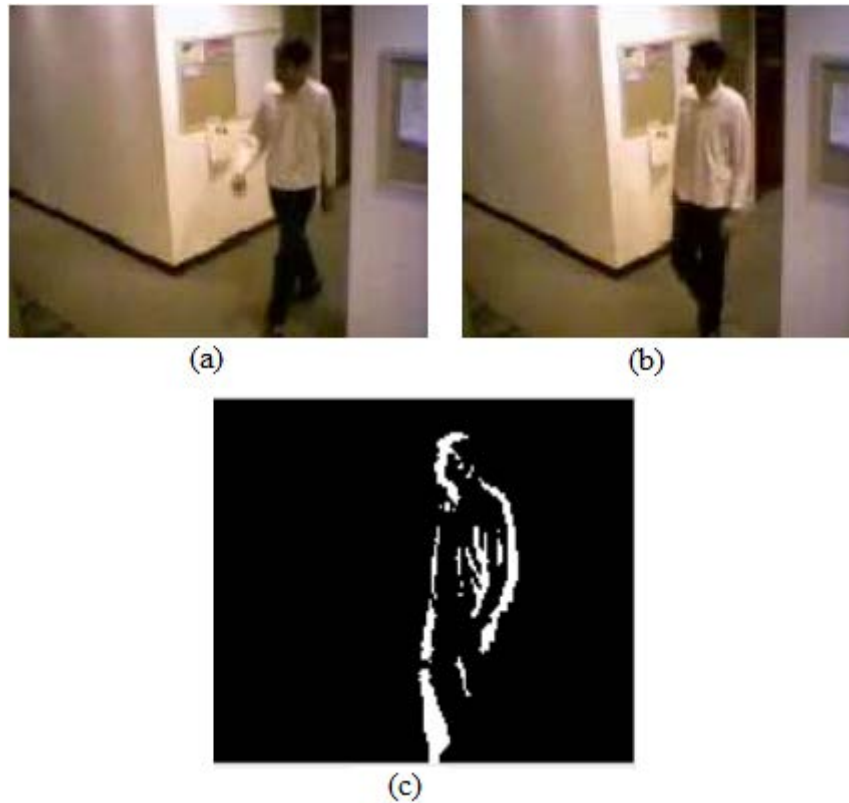


Fig.2.3 Frame difference approach, (a) the background frame (# $k-1$), (b) the current frame (# k) and (c) the difference image [29]

iii) **Temporal differencing:** Temporal differencing is also named as double difference. It is the suitable approach for scenarios in the case of moving cameras [31]. This technique computes the differences between two or three consecutive frames pixel-by-pixel in order to reveal shifting areas in the scene [34]. In the case of a moving camera, the motion of the target and the camera are jumbled up. This approach is easy and simple to implement, widely used, rapidly to adapt to the changes in lighting, camera movement, or dynamic environments, and yields accurate objects' motion. However, it's unsuccessful to detect the interfering areas of moving objects, ineffective to find stopped objects in the scene, and objects that conserve uniform regions. This approach takes a long time for calculations, and uses a lot of memory [28], [29], [34]. Furthermore, it becomes harder to detect an object moving towards or away from the camera, and the results might not be accurate [29], [31], [36], [37], [38]. *Temporal frame differencing* is illustrated in Fig.2.4.

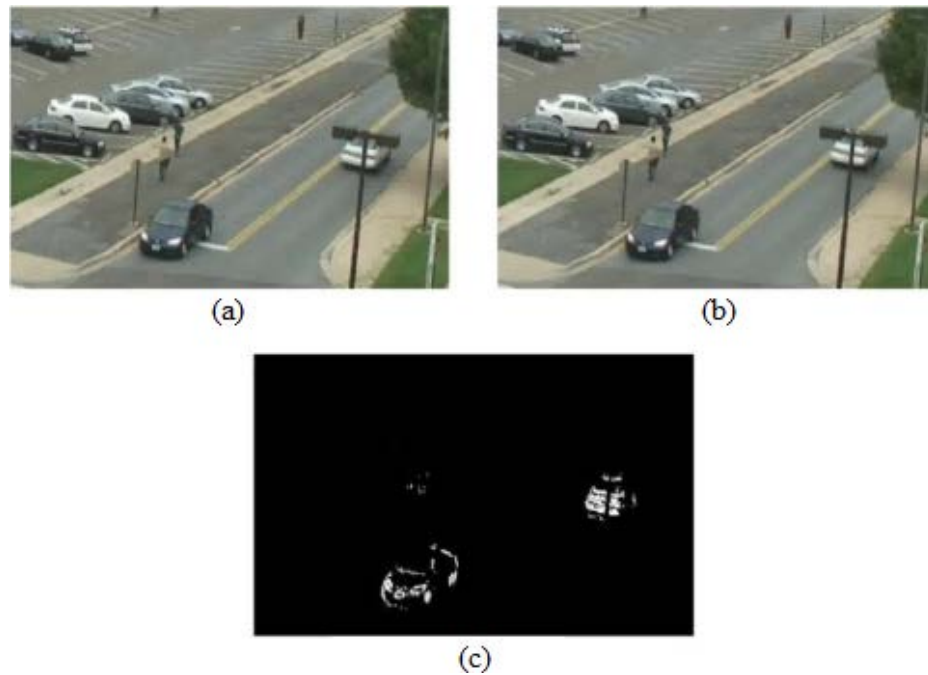


Fig.2.4 Temporal frame differencing sample, (a) Previous Frame(# $k-1$), (b) current Frame (# k), (c) Absolute frame difference result [34]

iv) **Optical flow:** It is a different technique for finding moving objects in video sequences. The apparent motion (i.e. the rate at which light patterns move) of illumination patterns inside a picture framework is referred to as optical flow [34]. This approach produces a two-dimensional vector, which represents the velocities and orientations of every pixel in successive image sequences. Discontinuities in the optical flow aid in partitioning images into areas that match with various objects. Although the approach is computationally costly and sensitive to noise, it has the advantage of being able to detect motion in video image series with multiple moving objects and dynamic backgrounds [30], [31], [34], [35], [37]. Fig. 2.5 shows a picture with its visual stream vectors.

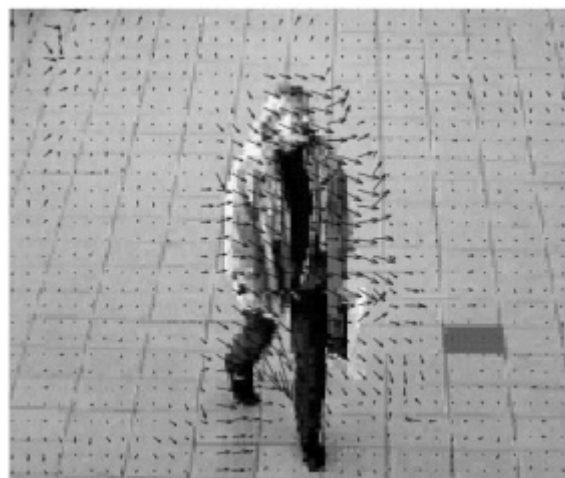


Fig. 2.5 Optical flow vectors in moving object [34]

- v) **Kalman Filtering:** This is a widely used signal-processing approach of recursive estimation for predicting the position of a moving object based on previous motion data.

2.1.2 Object modeling and representation

One of the significant aspects that define the performance of object tracking algorithms is object representation [39]. Object representation can be applied as the second step in the tracking process, which refers to the way of modeling and representing the tracked object. The tracked object is a sub-image (rectangular window) that includes the image of the moving object and its local background. The moving object might be vehicles, humans, birds, and/or any other type of moving object. Object representation is the approach of demonstrating objects. Therefore, to represent the object, some features that define the object should be extracted. The approach of demonstrating the scene's objects of interest is known as object representation. Representing the object means that some descriptors or features should be extracted/calculated. After that, these features are used to track the object during the course of tracking [29], [31], [32]. There are many techniques that could be used to classify object representation, such as those based on shape, color, texture, and motion [28], [29], [32]. Fig. 2.6 shows the classification methods to represent moving objects.

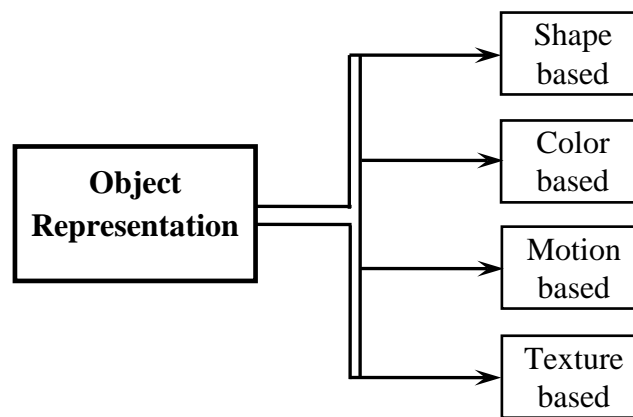


Fig. 2.6 Objects classification methods [29], [31], [32]

2.1.2.1 Shape based representation

Representation using shape is carried out by analyzing the geometrical shape and finding objects of similar shapes by matching with the contents of a relevant database. The contrasted images, form information about motion regions which can be obtained for classifying movable objects such as [4], [3]:

- **Points:** The object is represented by a single point, such as the centroid or a group of points; this method is appropriate for the tracked objects that take small areas of an image.
- **Primitive geometric shapes:** The shape of the object is represented by means of a square, rectangle, ellipse, etc. Simple, solid objects, as well as non-rigid ones, are better to represent them by primitive geometric shapes.
- **Object silhouette:** it is represented by the region inside the border of an object. This form is appropriate for monitoring and tracking non-rigid, complicated shapes.

- **Articulated shapes:** Articulated objects are comprised of body parts that are connected with joints, such as the human body, where all body parts such as the head, hands, legs, and feet are connected by joints.
- **Skeletal models:** Object skeleton can be obtained by applying the medial axis converted to the object's silhouette. This model could be used for representing rigid and articulated objects.

Every point in every frame is taken into account for classification, and results are kept as histograms. Pattern matching approach manner can be applied, but it does not perform well in a dynamic situation [28], [29], [32]. Fig. 2.7 shows some examples of object shape representations.

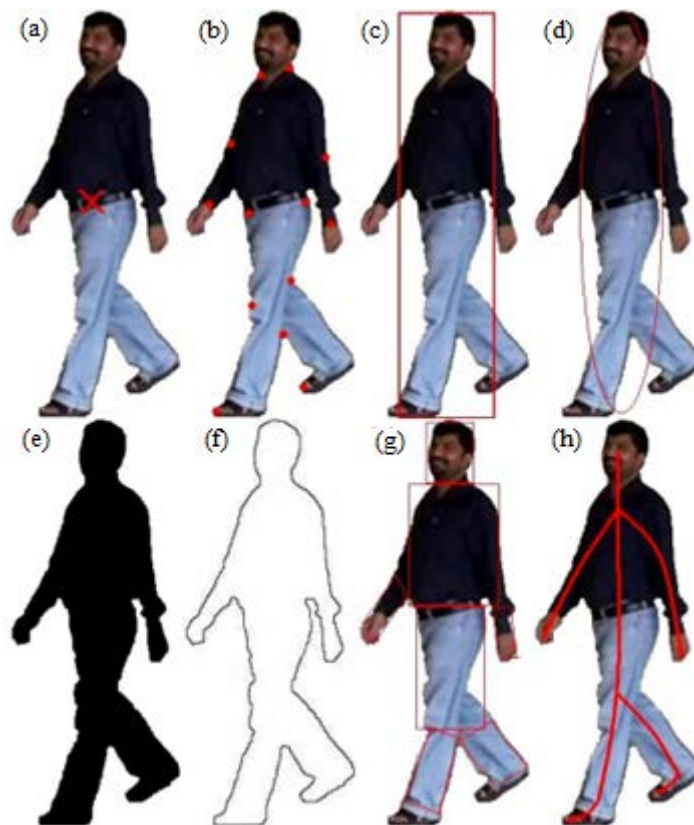


Fig. 2.7 Object representations [39] (a) point or centroid (b) multiple points (c) rectangular shape (d) elliptical shape (e) object silhouette (f) object contour (g) articulated shape (h) skeletal mode= object skeleton

2.1.2.2 Color based representation

In the case of the colored image, then the classification that relies on color could be applied to represent objects by utilizing the feature as color, which is a relatively fixed feature and doesn't change with the changing of the object view side and it is easy to obtain. In real-time applications, a technique relying on the color's histogram is used in object tracking processes [28], [29], [32]. A variety of color spaces can be used in representing information about colors such as RGB (red, green, blue), HSV (hue, saturation, value), and YCbCr (luminance and chrominance). But the RGB color space is mostly used [15]. Fig. 2.8 shows an example illustrating the original image in RGB space (Fig. 2.8.a) converted to HSV color space (Fig. 2.8.b), whereas Fig. 2.9 shows two different

images: Flower and Kingfisher as examples representing RGB color space (left) and YCbCr color space (right).

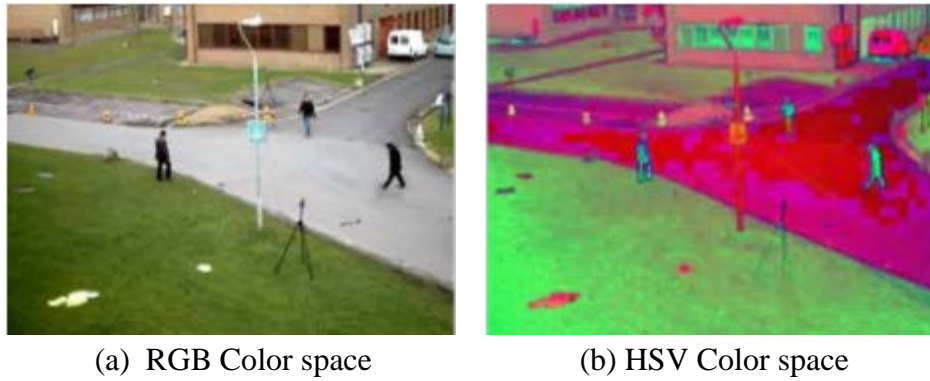


Fig. 2.8 RGB image (original) color converted to HSV space [40]

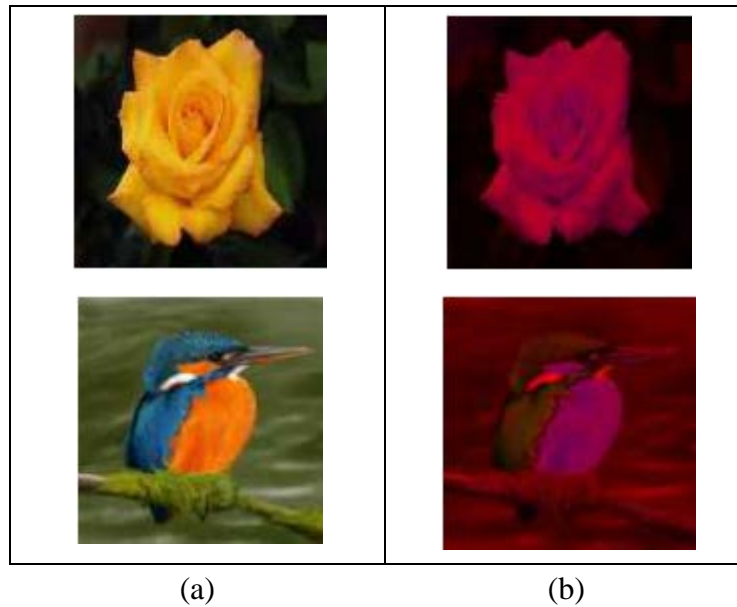


Fig. 2.9 Flower and Kingfisher Image in (a) RGB space, and (b) YCbCr Color spaces [41]

Histograms for the first component of RGB (R) and YCbCr (Y) color spaces are shown as samples in Fig. 2. 10.

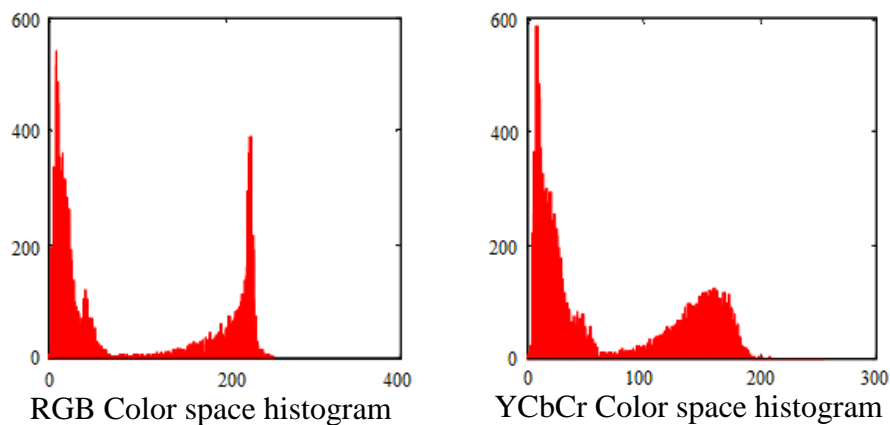


Fig. 2. 10 Histograms of the first component of RGB and YCbCr color spaces [41]

2.1.2.3 Texture based representation

The texture is an important feature for defining the characteristics of the images. It is a gauge of the intensity disparity of surfaces that defines the regular patterns and inherent properties in an image, such as uniformity, regularity, linearity, smoothness, density, directionality, and coarseness [28], [29], [32], [42]. As examples, some texture characteristics are demonstrated in Fig. 2.11.

The texture is an inherent feature and it describes the spatial ordering of density and concentration values or color in an image, which may be cyclic, stochastic, or both [17]. Two different types of periodical and stochastic textures are illustrated in Fig. 2.12 as examples.

The texture could be considered as a regional representation that is appropriate when the center of attention is internal properties [42].

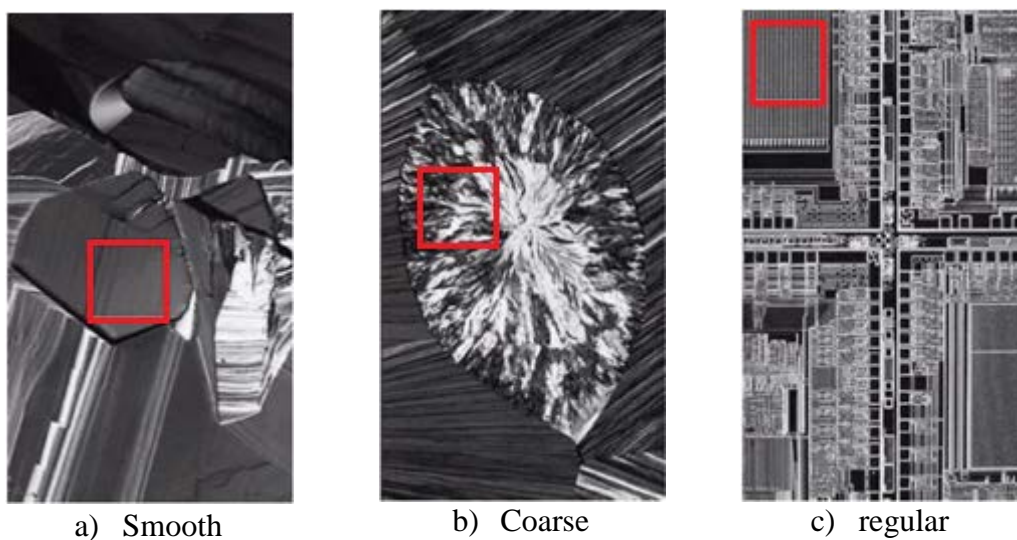


Fig. 2.11 The red box mark textures for optical microscope images of a) a superconductor, b) human cholesterol, and c) a microprocessor [42] R.C. Gonzalez and R. E. Woods: Digital Image Processing.

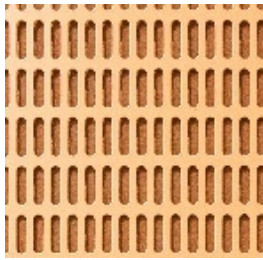
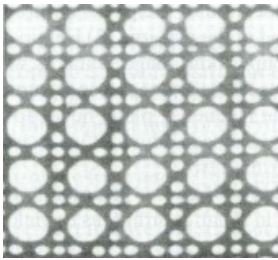


Periodic Texture		
Stochastic Texture		

Fig. 2.12 Images of two different types of periodic and stochastic textures

2.1.2.4 Motion based representation

Motion representation needs to take into account both solid and non-solid objects. Solid objects are easier to track than non-rigid ones since their motion is more predictable. Inflexible entities are likely to describe the short-term flow, whereas nonrigid items, such as humans, have a composite average residual flow with an interrupted component. Using motion-based categorization does not demand a predefined pattern template, it is hard to define non-moving humans [28], [29], [32].

2.1.3 Object tracking

Object tracking refers to the process of following a specific target of interest, or multiple targets, in a given video sequence [31]. It is achieved by producing or estimating the route for the moving object by finding its position in each frame through the object model formed in the previous frame as it moves in the scene [28], [29]. The major factors that make the tracking process in image processing a difficult task is the occlusion, loss of information, the non-uniform shape of the object, complicated object movement, noise in an image, and real-time requirements. Some types of the main approaches widely used to track a moving object are:

- Kernel tracking,
- Point tracking,
- Shape-based tracking,
- Feature-based,
- Motion-based tracking,
- Mean shift tracking,
- CAM-Shift tracking, and
- Silhouette tracking, etc. [28], [29], [32].

This thesis was focused on traffic applications, and the main points of interest were the general tracking strategy, the tracked object's appearance model, its motion model, overcoming troubles caused by common disturbances such as image clutter, shadows, and occlusions, as well as the concepts of fusion of obtained data in various ways.

2.2 Tracking strategies

The tracking process techniques for any specified object vary, relying on the required goals. For example:

- i) The objects that have specific characteristics, such as vehicles, people, faces;
- ii) Objects that have specific characteristics with a certain feature, such as movement of vehicles and people, and the face of a specific person;
- iii) Objects of unknown nature beforehand, but of specific interest, e.g. moving objects in general.

In each example (scenario), a portion (window) of the input frame is compared with a reference model that represents the object's appearance. Image patches, which characterize the appearance of

the tracked window at the intensity level of the pixel, shape, and/or general features such as color, edge, and/or texture can all be used as a reference [43].

In the last few decades, significant leaps in computer vision-based object tracking have been made. In general, the first step in the tracking of the object in the image sequence is that the object undergoes an initialization process where the tracked object parameters are specified, and it is centered in a bounding box in the first frame [24].

The mathematical exemplification of this window can be manifested as the features that describe it. Then the object description model is constructed based on extracted features that are utilized in the later frames.

The approach applied to estimate the state and foretell the track of the moving object in the subsequent frames is the tracking strategy (e.g., Markov chain Monte Carlo algorithm, Kalman filters, particle filters, optical flow algorithm, Normalized Cross-Correlation, and Mean-Shift, etc.) [25].

During the tracking process, the object status and surroundings are continually changing; this makes the extraction of features and building the model in the tracking process more difficult, which needs a more powerful tracker [25].

Since a video sequence contains a huge amount of data, it is crucial to extract only a small amount of significant information from computer vision-based object detection and tracking. According to the current computer vision literature, there are many methods used for object tracking techniques in a video sequence, but all have some disadvantages. [10], [44]. Some types of these methods are: model-based, feature-based, region-based, and contour-based tracking.

2.2.1 Model-based tracking

The model-based tracking method focuses on regaining high-accuracy trajectories and models of the tracked objects, and it relies on the appearance and shape of tracking objects in the estimation of an object's position [15], [45]. According to [46], a model-based technique requires a model for each tracked object (for example, a color model or a contour model).

This approach usually supplies a more robust solution. The main merit of the model-based methods is that knowing information about the scene helps in the refinement of performance and robustness through the ability to predict hidden object movement, discover partial occlusions, and act to minify the effects of disturbances generated during the tracking process [47]. The weak point of this method is the dependence on the geometric shape details of the object. It is impractical and/or un-expectable to have detailed models for every vehicle that might be seen on the road [10].

2.2.2 Region-based tracking

In this method, a sub-region of the frame is created as a bounded geometric shape (e.i. box or ellipse) around the initially known position of the moving object. This created an area (sub-region) is considered as the tracked object, which encompasses both the vehicle in the road and the surrounding portion of the background (road and area beside the road). The main principle here is to use more information with the vehicle's information, providing in this way a greater robustness [27]. There are considerable approaches that could be used for representing and modeling image regions. Mostly, the region modeling relies on the use of a probability density distribution of the

color, so the object is represented based on color. Color distribution could be described by a color histogram, or a mixture of Gaussian kernels [27].

This strategy acts properly well in free-flowing traffic [10] and is effective computationally, but it is ineffective when a number of objects move simultaneously in the image sequence, and it cannot track the moving objects accurately if they are in an occlusion situation [44]. Fig. 2.13 displays tracking regions in two images as examples.

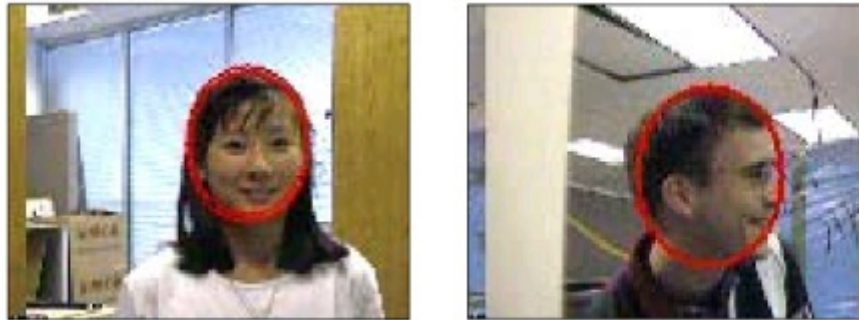


Fig. 2.13 Tracking region in images [48]

2.2.3 Contour based tracking strategy

This strategy depends on the extraction of an object's contour [28], [39], [49]. It is a discriminative method because it requires some form of picture apportionment to distinguish the object as a part of the image. The main goal is to obtain a representation of the bounded border of the object.

When the bounded border of the object is specified, its representation becomes the data and information of interest in subsequent frames. The advantage of a border representation instead of a region representation consists in decreasing computational complexity. However, the inability to separate the vehicles that are in a partial occlusion situation and the case of multiple vehicles in the region grouped as one object is the most important disadvantage of this approach [10], [15], [49]. In the contour tracking method, a contour that exemplifies an object in the previous frame repeatedly develops to the new location in the present frame. This developed contour demands a part of the object in the present frame be overlaid with the object area in the prior frame [50].

Fig. 2.14 illustrates an example of a head tracking result in rotation and movement case.



Fig. 2.14 Head tracking result using contour-based tracking [27]

2.2.4 Feature-based tracking

This strategy consists of the calculation of sub-features of a particular image which characterize the object of interest, like color, edge, points, corners, or texture to be used for the tracking purpose instead of tracking the entire region as a whole [10], [15], [51]. The most basic example is that the region of concern is represented as a collection of pixels of various light intensities and colors, where the same pixel pattern is searched in successive frames. As a result, the correlation-based technique would be a method with an extended time-consuming. Derived properties such as color distribution, border-point orientations, corner placements, and texture are more frequently used to define the region of interest [15]. According to [46], the feature-based approach utilizes visible features like the histogram of Oriented Gradients (HOG) [Dalal and Triggs, 2005], Haar [Viola and Jones, 2003] features to follow the observed objects.

The advantage of using features-based tracking is that, even in the existence of partial occlusion, some of the features of the object in motion stay obvious. Besides, the same algorithm can be applied for tracking in daylight or night-time at different conditions [3], [4]. In this work, the object tracking is represented as a rectangular region, surrounding the object and encompassing both features of the object itself as well as of the local background. The region-feature-based technique has been selected as the most effective one for our needs.

Fig. 2.15 illustrates examples of corner features-based tracking. Fig. 2.15.a shows a sample of corner features picked out by the tracker. The temporal development of many corner features in the image plane is shown in Fig. 2.15.b. When corner features reach the exit region, the grouping module groups them into vehicle hypotheses, as shown in Fig. 2.15.c [10].



(a)



(b)



(c)

Fig. 2.15 a) Tracker-detected corner features, b) A sample of the tracker's feature tracks, c) The sample of feature grouping [10]

2.3 Appearance model in object tracking

Defining the appearance characterization of the moving object is an almost significant indication for the tracking in video sequences [23]. The appearance model could be related to the approach of how the tracked target itself is represented, as well as the approach of how features have been extracted as a description of a region of interest and the approach to how they are formally represented.

There are the three fundamental representations for visual cues that could be used to portray the target's appearance: A 2D-array such as picture data, a 1D-histogram of ordered characteristics, or a feature vector. The appearance representation requires the transfer-ability of certain properties from one frame to the next as shown in Fig. 2.16.

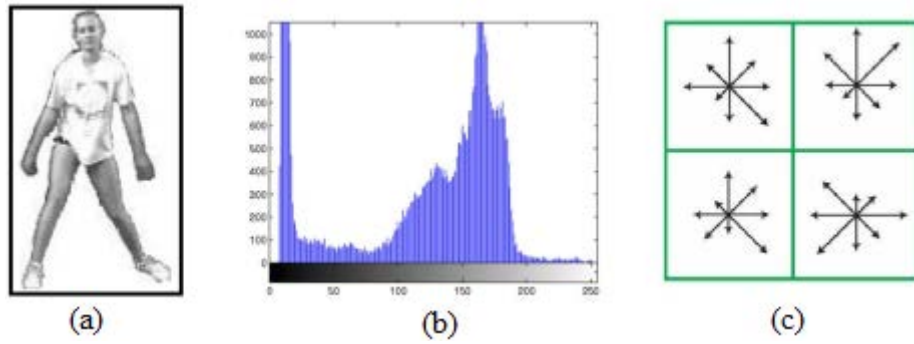


Fig. 2.16 Appearance representation. (a) 2D-Array, (b) Histogram, (c) Feature vector [52]

The use of color distribution to characterize the region was historically the first technique utilized [16]. Color-based trackers are fairly resilient in terms of object deformations, but they are extremely sensitive to changes in illumination, which may be compensated for by some kind of model adaption [53]. Several publications in this field are concerned with some form of object/region representation based on picture contrast, which leads to the extraction of contour points/corners. The calculation of picture gradients is an essential step in this categorization. For detection and tracking moving objects, many techniques based on the Harris Corner Detector – HCD [54], Scale Invariant Feature Transform – SIFT [55], and Speeded-Up Robust Features – SURF [56] are used. A typical feature for detecting a human in an image is the Histogram of Oriented Gradient (HOG) [57], and this technique also could be used to describe the region contents of interest, and the coherence of an object's appearance in successive frames is used to track it [46].

An additional feature of an image that can be used to describe the contents of a region is a texture (Local Binary Pattern – LBP, Grey Level Co-occurrence Matrix – GLCM, Harrallic texture features,...). Although this feature hasn't been used very frequently in tracking applications, there are several examples [15], [58].

In the context of processing a series of images, the object's appearance can be described by a bounded rectangular window, or an elliptical shape (image template as an example) with feature descriptors, that are represented as a histogram of color, or edge, or texture, or histogram of oriented gradient (HOG), etc.

Fig. 2.17 and Fig. 2.18 illustrate the image template and some histograms which represent the object appearance on the given frames as examples.

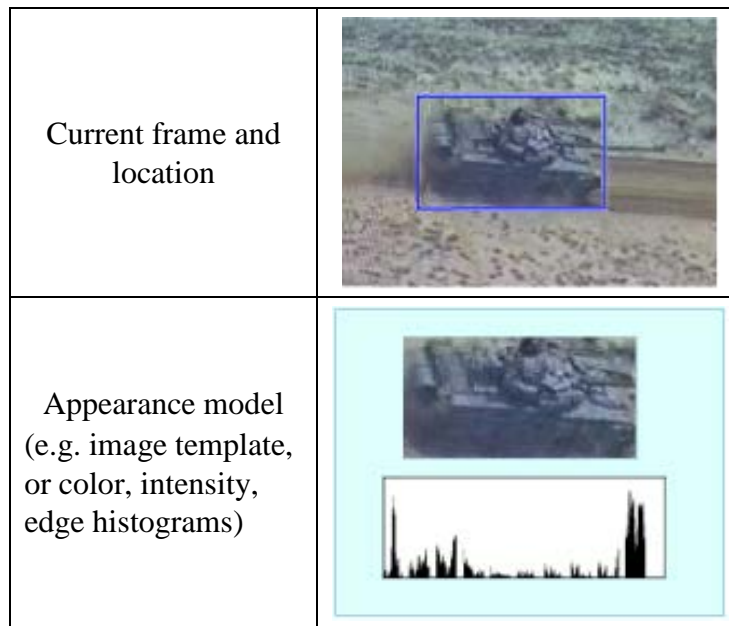


Fig. 2.17 Appearance model: Image template, color, or intensity, or edge histograms [48]

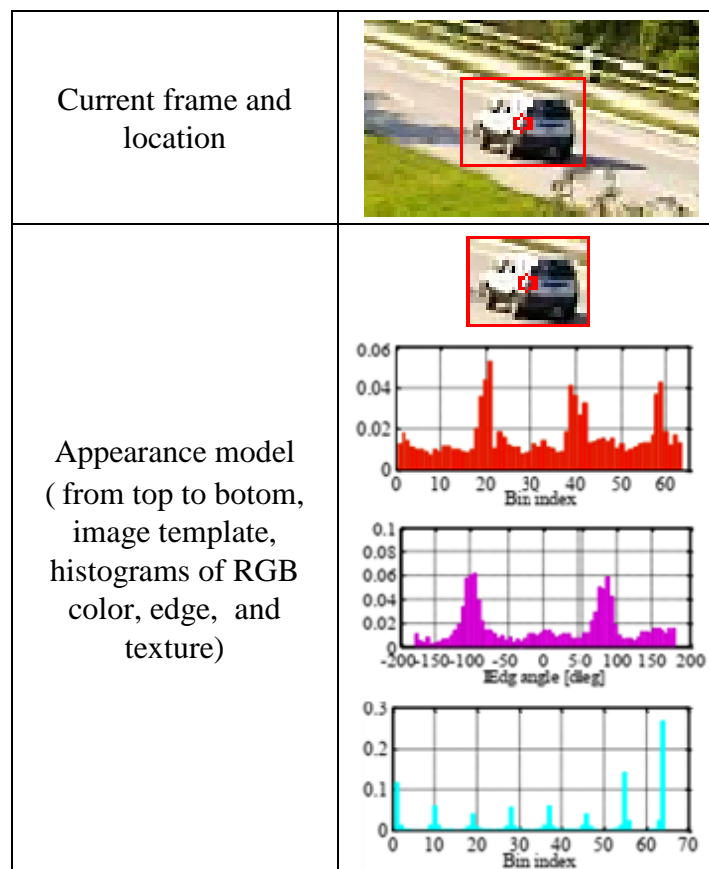


Fig. 2.18 Appearance model: Image template, RGB color, edge, and texture histograms [15]

2.4 Motion model

Object tracking aims to determine or estimate an object's location within subsequent video image sequences based on previous scene information. This principle can be accomplished by predicting the moving object's position in video image sequences based on the motion model. The motion model terms the potential motion of the object between consecutive frames, and it is used to specify the region in which a given target is expected to appear in the next image [59]. The motion of the object is usually in parametric forms, such as translation, rotation, and affine movement, which are used to describe how an object moves. The appearance representations or methods utilized to estimate the object's mobility vary based on the visual tracking techniques used in this approach [50]. The physical properties of the object's motion must coincide with the object's motion model.

The vehicular motion along the road (reference coordinate frame) has been assumed as a planar and translational one. The additional assumption is that the magnitude and orientation of a velocity vector defining the motion are constant and are useful under the assumption that the period between measurements is tiny. If these intervals are extended due to the occlusion of an object, the prediction capabilities of the filter in use become important. Equation 2.3 terms the motion or state model, which is defined as a function of the preceding state X_{k-1} and a noise variable n_k .

$$X_k = f(X_{k-1}, n_k) \quad (2.3)$$

The image information and data at the present time step k are utilized to calculate the measurements Z_k in this system. Observations on the location and appearance of the target are obtained via feature extraction computations. The measurement function of the present state X_k and a measurement error v_k at time k form the observation model, which links measurements to states [60].

$$Z_k = h(X_k, v_k) \quad (2.4)$$

Traditionally, the Kalman filtering concept has been used in vehicle tracking as the first approach [61]. The need for explicit measurements of model states, for example, the position of an object's centroid, velocity, also the assumption of the Gaussian nature of measurement errors and state uncertainty, made this technique to be less attractive, despite the fact that it is still used [62]. Particle filtering (PF) makes use of Spatio-temporal information about the target item rather than assuming perfectly known dynamic behavior or a motion model with a given level of uncertainty. That is why PF is preferred in real-world tracking applications [1], [24], [63]. This is a built-in advantage that allows the PF to function well in the face of typical image disruptions [15].

2.5 Disturbances

The presence of fluctuating shadows and reflections, as well as the occurrence of partial or even full occlusion of the tracked object, are the most common disturbances in video tracking of road vehicles. In these situations, there are a few basic measures to follow:

- 1) the presence of these irregular working conditions should be recognized based on the shadow model [64], or some indicators that discriminate the occlusion;
- 2) the proper action for this working regime should be taken, such as by modifying the distribution of motion parameters [24], calculating and analyzing the probability of object presence in a certain image region [65], attempting to discover non-occluded sub-regions [51], and so on;
- 3) The tracking system should be able to detect when the abnormal circumstance is no longer present, commonly based on some assumptions about the object's conduct during the occlusion [15], [66].

2.6 Choice of a Particle Filter (PF)

Kalman filter (optimal state estimator) is a common tool for target tracking in the probabilistic framework, but it cannot accurately dissolve the tracking problem if the model is nonlinear and non-Gaussian and if it is not well approximated [67].

A particle filter is a suitable tool for estimating the states of moving objects due to its good performance, its ability to solve the problems of clutter and occlusion, and the fact that it is robust for solving the problems of non-linear, non-Gaussian in dynamic state estimation as mentioned in many kinds of literature. In general, PF is a hypothesis tracker that uses recursively a set of randomly sampled particles of weights ' w_t^i ', to build and approximate the posterior possibility density function (PDF) using the Monte Carlo techniques, and computes estimates based on these samples and weights [25], [67], [68]. Therefore, Particle filters could be considered as a substantial alternate to Kalman filters, which are the most appropriate for non-linear systems and linear systems that suffer from Gaussian noise [49]. Estimating the state of the moving object of interest is estimated by the weighted sum of particles. Prediction and updating are the two primary phases in the tracking process. In the prediction step, each particle is altered in accordance with the state model of the region of interest in the video frame, which includes the insertion of random noise to imitate state noise. The weight of each particle is re-evaluated with the newly observed data in the update phase. After that, a resampling technique removes particles with low weights and duplicates those with higher weights [25].

Numerous of the proposed particle filters for video sequence tracking are based on a single feature, such as color, texture, or edge gradients, etc. but, when clutter exists in the background, single-feature tracking may not always deliver dependable performance. Thus, using multiple-features based PF for tracking the moving object improves the robustness and provides a richer description of the object [25].

2.7 Fusion of multiple cues

A fusion of several cues is an efficient strategy to overcome the specific shortcomings of some of the strategies outlined above. Fusing more than one feature would drive the vehicle tracking process to be more accurate and efficient. The necessary data and information could be acquired by different sensors combining color and infrared images [69], as well as by different color channels [12], or by combining color and segmentation information [70], or by combining other features. The texture of the region could be combined with depth information [71]. The multi-feature merging method is largely significant and effective in adaptability capacity, especially in the presence of disturbances [15], [18], [26] it is espoused in this thesis as well.

Chapter 3

Tracking of Moving Object Based on Multiple Image Features Using Particle Filter

3 Tracking of Moving Object Based on Multiple Image Features Using Particle Filter

3.1 Particle Filter Applied in Video Object Tracking

Any type of visual tracking of moving objects starts by discovering the object to be tracked. This step can be done automatically or by a human operator. After that, the process should be automatic from this point on. The tracking algorithm foretells the next position of an object depending on the assumed model of motion (2D – 3D, continuous – with stops, maneuverable – non-maneuverable, etc.). The estimated position of the object is revised once a new set of measurements is taken, and the procedure is continued. The way how the set of previous measurements and estimates are used in the following prediction varies between different filters/estimators (data block processing, iterative ones – based on balancing between state and measurement uncertainties, based on maximal likelihood, etc.)[15]. Visual tracking of moving objects could be implemented and achieved by many approaches as mentioned in the previous chapters, and in some of the computer vision literature. From a tracking algorithm standpoint, the choice made in this thesis was a “particle filter” (PF), which is one of the available techniques that could be used.

Particle filters, also known as Sequential Monte Carlo approaches, use a set of random particles with associated weights to approximate the Bayesian posterior probability density function (PDF) and compute estimates based on these samples and weights. The main notion behind particle filters is that rather than finding a precise representation of a simplified model (Gaussian), one can create an approximate representation of a complicated model (of any arbitrary Probability Distribution Function – PDF).

The main reasons for using PF were:

1. It is a helpful tool that can be used in a diversity of situations:
 - Applicable in the “Image processing” context;
 - Both for automatic detecting and tracking of objects on the scene;
 - Capable of coping with complex environments.
2. It allows the analysis of complex systems without the need for their simplification:
 - Non-linear
 - Non-Gaussian

The Kalman filter was the main approach for solving state-space models before particle filtering techniques gained popularity. A linear Gaussian state-space model can be solved with the Kalman filter. Its versions, the extended Kalman filter, and the unscented Kalman filter, can be utilized when the linearity or Gaussian criteria are not met. However, they fail to offer a realistic estimate for extremely nonlinear and non-Gaussian cases[72].

Modelings of non-linear and non-Gaussian systems are essential and required for tracking objects in video. How can we acquire a solution for tracking a moving object if we have to include these

two phenomena? A probabilistic approach that constructs tracking as inference in a Hidden Markov Model (HMM) could be utilized as one solution [11]. Fig.3.1 shows how to make this frame.

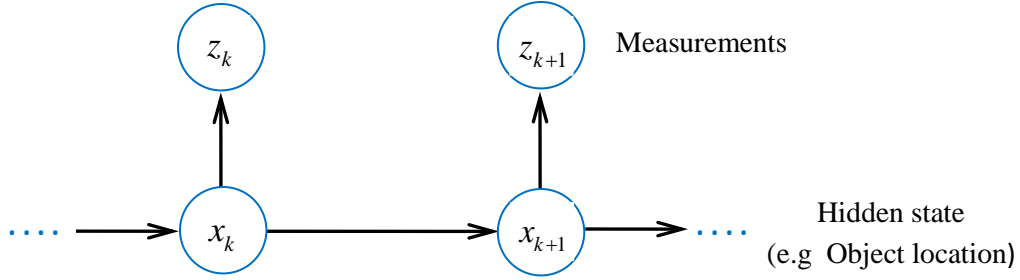


Fig.3.1 Probabilistic frame work for tracking system[11]

On figure 3.1:

x_1, x_2, \dots, x_{k+1} are hidden states (Markov chain), and

z_1, z_2, \dots, z_{k+1} are observations.

Particle filtering is an encouraging approach since it performs well with both Gaussian and non-Gaussian noise errors that are incorporated in the system measurements[73], and it utilizes and fuses multiple features such as color, edge, and texture to follow a single object or a group of objects in movement in video sequences. The PF is a robust technique to track objects in image sequences with complicated backdrops[68].

3.1.1 Bayesian's filter

The particle filter is essentially based on the Bayesian filter. It is therefore necessary to understand the Bayesian filter to comprehend the PF.

The Bayesian filter is based on Bayes' theorem premise. As shown in[73], [74], a nonlinear system can be specified and written as follows:

$$x_{k+1} = f_k(x_k, n_k) \quad (3.1)$$

$$z_k = h_k(x_k, v_k) \quad (3.2)$$

Where,

f_k : Transition equation (state equation) at time k ,

h_k : Observation equation (measurement equation) at time k ,

x_k : State of the object which is to be estimated,

z_k : System's observation (measurement),

n_k : Process noise (system white noise) induced into the system due to process,

v_k : Added noise to the measurement (measurement white noise) due to measuring instruments,
 k : Time index.

A Bayesian estimator's purpose is to approximate a conditional PDF, of the state x_k given the measurements z_1, z_2, \dots, z_k . This conditional PDF is indicated as $p(x_k / z_{1:k})$.

At the beginning of the iteration process, the initial measurement is taken at $k = 1$. Consequently, the estimator's initial condition is the PDF of x_0 , which can be represented as [73], [74]:

$$p(x_0) = p(x_0 / Z_0) \quad (3.3)$$

Where Z_0 : The set of initialized measurements at $k = 0$.

If we assume that there are two variables x and z , the $p(x, z) = p(z, x)$ is called Joint commutative probability, which is relied on the Bayes theorem [75].

$$p(x / z)p(z) = p(z / x)p(x) \quad (3.4)$$

$$p(x / z) = \frac{p(z / x)p(x)}{p(z)} \quad (3.5)$$

Equation 3.5 represents the posterior PDF, it can be written as:

$$posterior = \frac{(likelihood)(prior)}{evidence} \quad (3.6)$$

3.1.2 Particle filtering

Particle filters were first offered by Nicolas Metropolis in 1949, who proposed analyzing systems by looking at the features of groups of particles rather than individual particles. However, computational and processing capability has only been enough for their execution in the 1980s. Even now, the particle filter's computational complexity is the main barrier to its spreading extent and use. The particle filter is a statistical technique which estimates the work effectively for issues that the traditional Kalman filter fails to solve. The particle filter is used to numerically execute the Bayesian filter [73], [74]. In the initialization step, dispersed random numbers of particles (N) for PDF $p(x_0)$ would be created. These particles are moved to the next step using the transition model. When the number of particles is raised, it leads to more accurate state estimation, but it also causes problems with calculation complexity. As a result, the number of particles selected is restricted to a particular number based on the intricacies, environment, and conditions of the system [73].

The fundamental particle filtering technique is based on applying relevant sampling techniques to update a distribution in a sequential way. The sequential importance sampling (SIS) method, developed by Kong et al., is a particle filtering technique (1994). SIS is a method of solving the recursion problem by applying importance sampling[72].

PF, in general, utilizes the Monte Carlo approach to recursively construct the posterior probability density function of the state space[43], [49], [67], [68]. The primary idea behind PF is to represent a posterior PDF through a set of weighted samples that are picked at random. Each particle represents a state hypothesis, and it is randomly spread based on the transition model from the previous density. The likelihood model is used to weigh each diffused particle, and the weight describes the particle's quality[9]. The posterior PDF is approximated recursively by a collection of weighted particles which are randomly sampled. The tracked object's state at the time \mathbf{k} is indicated by x_k and is represented by a collection of samples in motion space, while the system's observation is pointed out by z_k . Like a Bayesian filter, the PF performs two primary processes at each iteration: prediction and updating[9], [76].

- Prediction: In each frame \mathbf{k} , the samples are propagated depending on the system model (3.1), and the particles are spread independently. The prediction step gives an approximation of the previous PDF that corresponds to the propagation of particles x_k^i at the time $k-1$, and it is performed by equation (3.7):

$$p(x_k) \approx \frac{1}{N} \sum_{i=1}^N \delta(x_k - x_k^i) \quad (3.7)$$

Where N is the total number of particles and $\delta(x_k - x_k^i)$ is the Dirac delta function.

- Update: in this step, at time \mathbf{k} , the measurement z_k is available, the weight of each sample is calculated using the likelihood model (observation model (3.2)), and the discrete approximation of a posterior PDF $p(x_k / z_{1:k})$ at time \mathbf{k} is calculated using the equation (3.8):

$$p(x_k / z_{1:k}) \approx \sum_{i=1}^N \tilde{w}_k^i \delta(x_k - x_k^i) \quad (3.8)$$

where : \tilde{w}_k^i is normalized weight, and it is given as:

$$\tilde{w}_k^i = \frac{w_k^i}{\sum_{j=1}^N w_k^j} \quad (3.9)$$

so that, the normalized weight set satisfies:

$$\sum_{i=1}^N \tilde{w}_k^i = 1 \quad (3.10)$$

The particle weight w_k^i is presented by equation (3.11)

$$w_k^i \approx w_{k-1}^i \frac{p(z_k / x_k^i) \cdot p(x_k^i / x_{k-1}^i)}{q(x_k^i / x_{0:k-1}^i, z_{1:k})} \quad (3.11)$$

Where: $p(z_k / x_k^i)$ is the likelihood function, $p(x_k^i / x_{k-1}^i)$ is the state transition density distribution, and $q(x_k^i / x_{0:k-1}^i, z_{1:k})$ is the proposed distribution (importance distribution/function) in which the particles are samples.

Iterations of PF based tracking systems consist of four main steps:

- State transition (prediction): using the motion model,
- Measurements (Likelihood estimation): by observing the features using the observation likelihood model,
- State estimation: based on maximal particle weights,
- Resampling (Selection step): a selection of the new set of particles for the next iteration.

3.1.3 Particle filter algorithm

To make the strategic description of a single iteration of the recursive PF algorithm simpler, suggested tracking process steps are represented as follows:

1. Initialize the target state x_0 at $k = 0$ in the first frame, and calculate the reference histogram based on the proposed feature (color, edge, or texture) for the target window of the size $height \times width$.
2. Assume the initial state of the system PDF $p(x_0 / z_0)$ as initial distribution $p(x_0)$ is known. Generate a set of random particles ($x_0^i; i = 1 : N$) based on the $p(x_0 / z_0)$, around the target point.
3. Predict new particles' state at the time k to determine a prior particles' state, it is done by spreading each particle based on the transition model (equation (3.1)),
4. Compute histogram distance (Bhattacharyya distance for color, edge, or texture features) for each particle window,
5. Compute the weight w_k^i (equation (3.11)) for each particle for the measurement z_k based on histogram distance using the proposed features, and normalize the weights.
6. Select the location of the target as the particle with maximum weight (minimum histogram distance).
7. Resample the particles according to their weight to generate new samples for the next iteration,
8. Increase time step, go to step 3.

The general flow chart algorithm of the particle filtering process is shown in Fig.3.2 below.

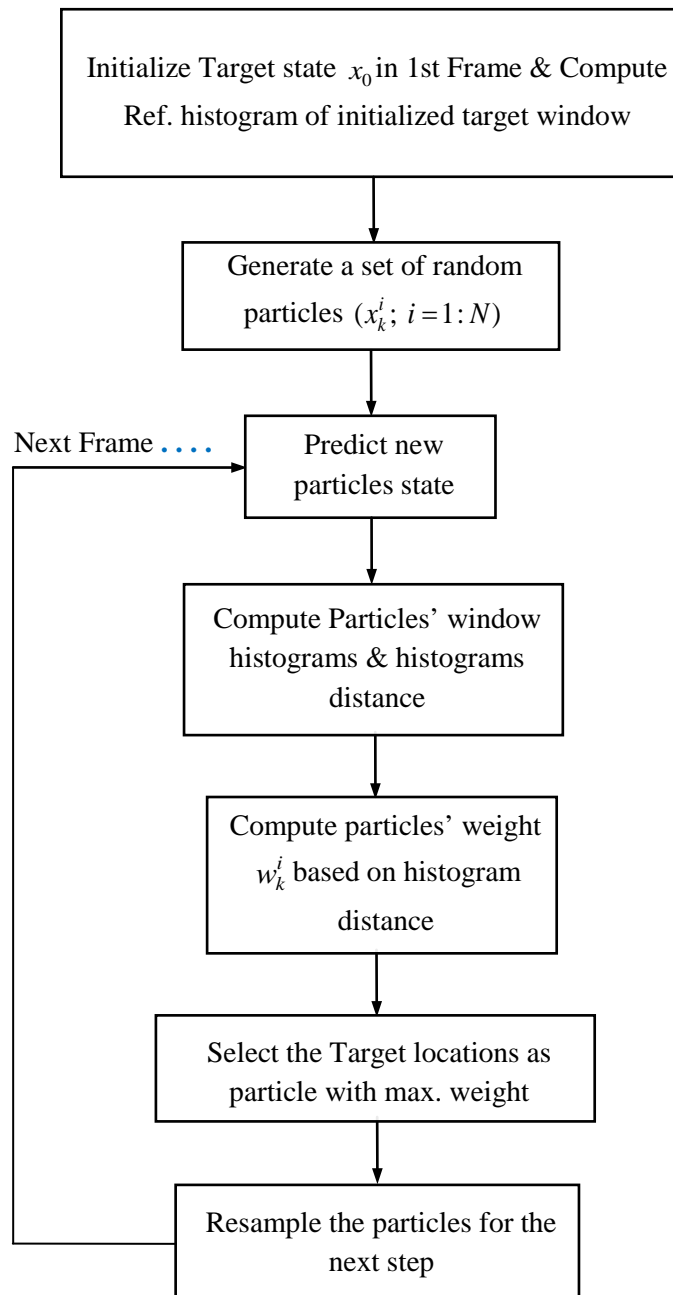


Fig.3.2 general particle filter flow chart

3.2 State Transition

The transition model/state motion model describes how objects move from one frame to the next. In this thesis, based on the premise that the motion of the object along the road is planar, translational, and 2-dimensional motion, and that the magnitude and direction of a velocity vector are constant in a short period between measurements, the following equation (3.12) describes the general transition state model[11], [15], [67]:

$$\vec{X}_k = \begin{bmatrix} x \\ y \\ V_x \\ V_y \end{bmatrix}_k = F \begin{bmatrix} x \\ y \\ V_x \\ V_y \end{bmatrix}_{k-1} + \vec{n}_k \quad (3.12)$$

Where: the state vector \vec{X}_k at time k in Image Coordinate Frame (ICF) comprises of position and velocity components,

F is the transition matrix,

x and V_x are position and velocity in x direction respectively, and

y and V_y , in the y direction, and

\vec{n}_k is a white Gaussian process noise of the system model with known PDF, and

Δx and Δy are the position increments computed in the previous step.

Fig.3.3 shows a schematic representation of this step.

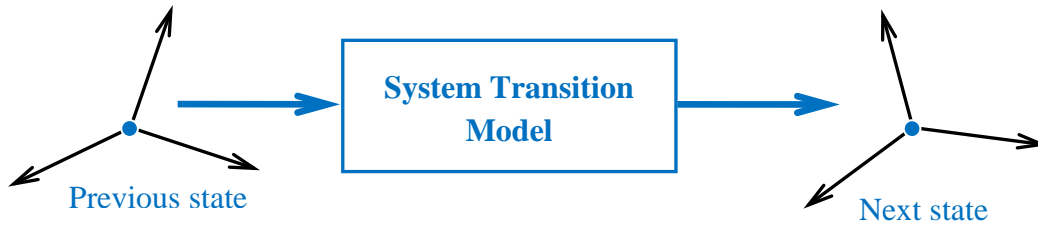


Fig.3.3 Transition model step representation

This transition model reduces to:

$$\vec{X}_k = \begin{bmatrix} x \\ y \end{bmatrix}_k = \begin{bmatrix} x \\ y \end{bmatrix}_{k-1} + \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}_{k-1} + \vec{n}_k \quad (3.13)$$

If the sampling intervals are rather long, followed by the vehicle accelerating/braking or performing angular maneuvers, the validity of the previously mentioned assumptions is jeopardized (particularly in the occlusion periods).

3.3 Visual Tracking Based on Image Features

In many approaches, the tracking processes are based on the tracking of the entire object, but tracking based on features tracks only the proposed features representing the window around moving objects from frame to frame. Representation of an image can be made by means of a single feature vector or sometimes a combination of features may be used to represent that image. This method has several advantages, such as the ability to see/recognize some of the moving object's features even when there is partial occlusion[77].

Since there is not enough information provided about the object being tracked, if a single feature is used for tracking, it is possible to lose track of the object and begin tracking the wrong one. Hence, if many features are employed, more information about the object can be provided.

From a tracking strategy point of view, the choice to apply a region/feature-based method was motivated by the expectations:

- Increasing the level of information if the tracked object is the vehicle accompanied by the surrounding area;
- Reduced amount of calculation if the features of the region of interest are calculated instead of the complete contents of it;
- Higher robustness if a few features are properly included and the results of their usage are integrated.

The feature tracker depends on the deterministic search of a window, whose feature histogram matches a reference window feature histogram [78].

In video object tracking, features can be easily identified in the feature space. The choosing of feature is primarily related to the exemplification and the particular type of the object (vehicles moving on the road, boats or ships in the sea, walking people in the street). Hence, choosing the right feature has a significant turn in the tracking process. The most usually used features for tracking in literature are those based on the color, shape, texture, and optical flow in the image [41], [77], [79]. In this work, features derived from the color, the object's contour, and the texture inside the window, is intensively considered in the context of tracking.

3.3.1 Color feature

In the literature, several approaches for tracking applications based on color attributes have been published. Color is a simple and straightforward feature to compute and implement. There are many color spaces that have been utilized in tracking applications, such as RGB space (red, green, blue), HSV space (hue, saturation, and value), YCbCr space (luminance and chrominance), etc. Color histograms are used extensively in the color feature extraction process, and a mathematical representation of the collection of colors is called color space. Below is a brief description of RGB, HSV, and YCbCr color spaces:

- **HSV Color Space:** Although RGB is the most popular, there is a multitude of color spaces to choose from. This is due to the fact that different color spaces convey and afford color data in different ways, making certain calculations easier and providing a more straightforward approach to

identifying colors. The HSV color space (Hue, Saturation, and Value) is much near at hand to the RGB color space. Hue (H) is the major color seen by humans, which is defined as the angle (in the range 0 to 2π) about the central axis. The saturation (S) is defined as the quantity of white light that varies (assorted) in color and it is stated as the radial distance started from the central axis to the outer surface in the range from 0 to 1. The value (V) represents the measure of brightness/intensity, which is the span along the central axis in the range from 0 to 100 [80],[81]. Fig.3.4 shows the HSV Color Space.

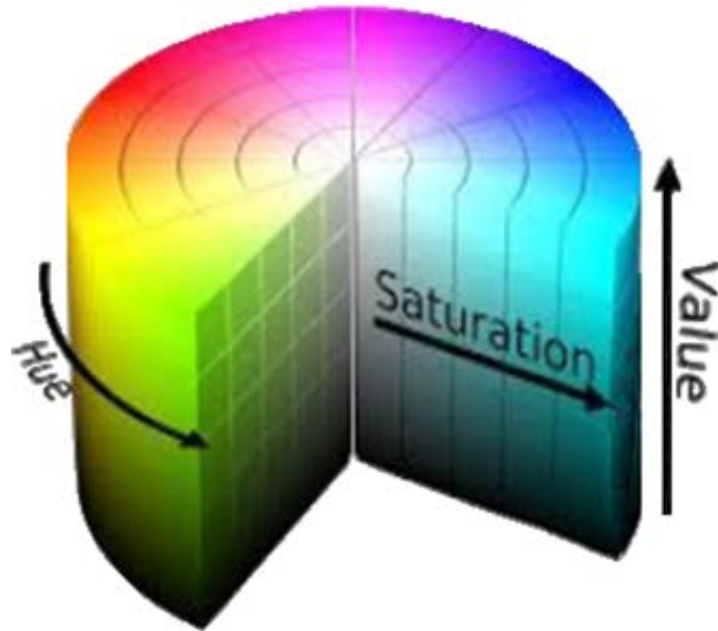


Fig.3.4 HSV Color Space[80]

- YCbCr Color Space:** The YCbCr color space can be seen in Fig.3.5. YcbCr, and RGB are distinguished by the fact that RGB represents color as a combination of red, green, and blue, whereas the YCbCr space appears the color as brightness and two additional color contrast signals. The Y in YCbCr stands for brightness (luma), Cb stands for blue minus luma (B-Y), and Cr stands for red minus luma (R-Y). The advantage of the human eye is being used to create this color space. Changes in light intensity are more noticeable to the eye than changes in color. When storage space is limited, the intensity component can be recorded with more precision than the Cb and Cr components[41].



Fig.3.5 YCbCrColor Space[41]

- **RGB Color Space:** RGB color space is depicted in Fig.3.6. The trichromatic theory is used to create this color space. Typically used in systems that display images using a cathode-ray tube (CRT). RGB space is widely utilized in computer systems, televisions, other electronic devices, as well as image processing[41].

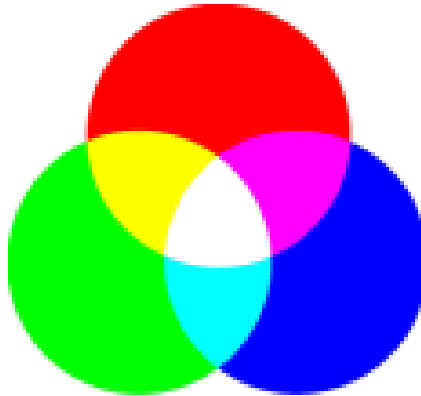


Fig.3.6 RGB Color Space[41]

The RGB color space is the most widely utilized. Local-global color histograms, color space partitioning, the use of a clustering technique, and other methods, can be used to represent colors[41]. The color histogram is the most commonly utilized for this purpose because of its invariance during a rotation or changing orientation and its robustness to scaling changes [41],[82]. On the other hand, it performs poorly when the lighting changes or the background is the same color as the tracked object[79]. The color histogram employed in our method is made up of three sequential histograms (R, G, and B), each having 21 bins, for a total of 63 bins. In the second column of Fig.3.7, typical color histograms for candidate windows in various frames from an actual sequence are displayed[15].

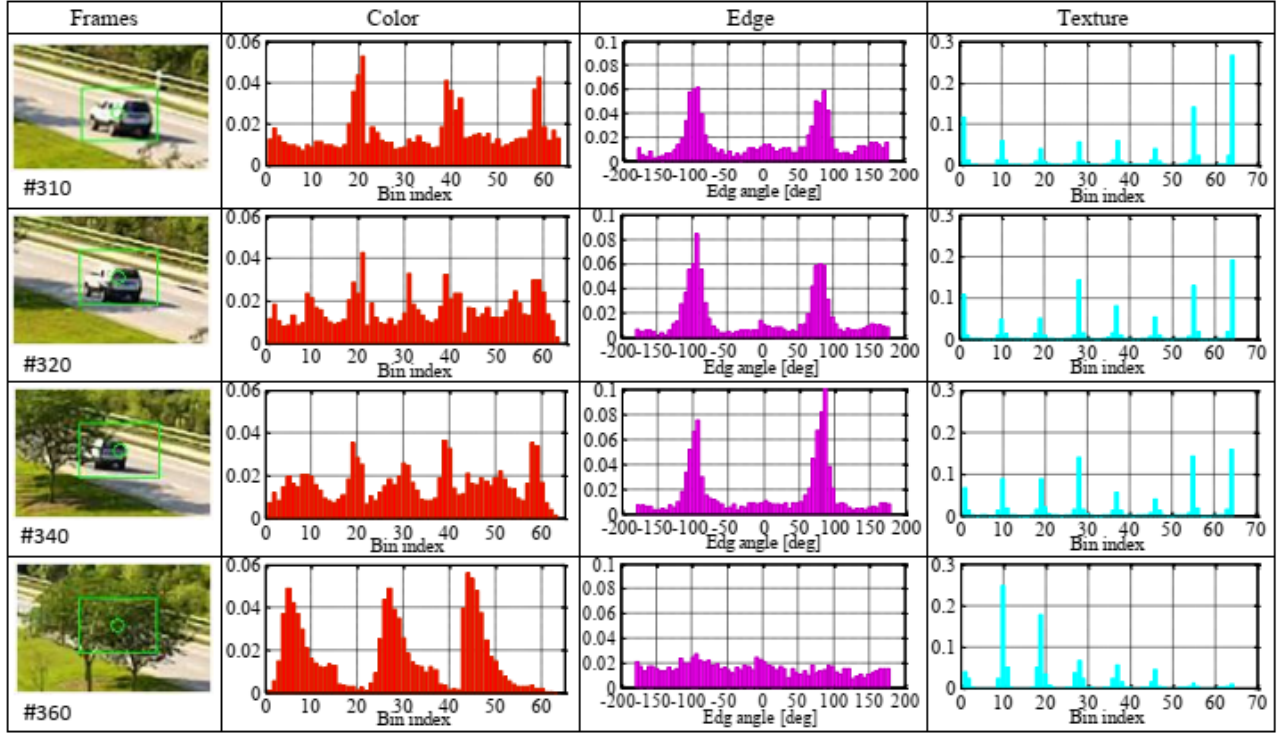


Fig.3.7 Color, Edge, and Texture features Histograms for the candidate windows in real video sequence (frames: #310 – initial (reference), #320 – with shadow, #340 – partial occlusion, and #360 – full partial occlusion)

3.3.2 Edge (contour/border) feature

The edge of an object is a collection of pixels having strong gradient intensities. The edge feature provides shape information that has been shown to be useful in visual contour tracking. In comparison to color features, the edge feature is less sensitive to variations in illumination[44]. This representation of the item's contour, on the other hand, is frequently sensitive to the presence of other pixels in the image that don't belong to the object but satisfy the high gradient intensity condition. The edge feature is defined in our work as a histogram that expresses the distribution of edge points' gradient directions. The edges of the object are distinguished using a variety of gradient-based operators (Sobel, Canny, Roberts, and Prewitt). In our method, the edges are extracted by estimating Sobel operators K_x and K_y on a greyscale image. The magnitudes of the horizontal gradient G_x and the vertical gradient G_y are determined as follows:

$$\begin{aligned} G_x(x, y) &= K_x \times I(x, y) \\ G_y(x, y) &= K_y \times I(x, y) \end{aligned} \quad (3.14)$$

$$\text{where, } K_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, K_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \text{ and } I(x, y) \text{ is the candidate window.}$$

The following formulas are used to compute the magnitude (G) and phase (θ) of gradients along edges:

$$\left. \begin{aligned} G(x, y) &= \sqrt{G_x(x, y)^2 + G_y(x, y)^2} \\ \theta(x, y) &= \tan^{-1} \left(\frac{G_y(x, y)}{G_x(x, y)} \right) \end{aligned} \right\} \quad (3.15)$$

The histogram of gradient phase angles is a distribution that describes the contour of an object. For the unsigned gradient, the overall range of phase angles is -180° to $+180^\circ$, divided into 64 equal bins. The steps for determining the phases of gradients in the histogram of contour points are as follows:

- Determine which pixels (positions) have a magnitude (G) that is greater than a certain threshold (e.g., 80 percent of maximal magnitude in this window).
- Determine the phase angle (θ) gradient that reconciles to the extracted pixels.
- Produce the histogram of these phase angle gradients.

The third column in Fig.3.7 shows the histograms of phase angle gradients for the chosen windows.

3.3.3 Texture feature

The texture is a grade of the consistency level and change in intensity that defines characteristics such as regularity, linearity, uniformity, density, directionality, smoothness, and coarseness of a surface[42]. The texture is known as the spatial configuration of intensity values or color in an image, which could be periodic, stochastic, or both[17]. There are several methods employed for describing and representing texture, including structural, statistical, and spectral approaches[83]. We chose the statistical techniques and put three of them to the test:

- i) **Local Binary Patterns (LBP):** It is an approach used in many applications. LBP is a grey-level measure used to characterize surface texture attributes. It's a robust and dependable method for describing a texture's pure local binary based on the texture unit[84],[85]. A texture unit (TU) is implemented by eight elements. Each element has one of two potential values (0,1) derived from a 3×3 local neighborhood in the two-level version. Only $2^8=256$ texture units are potential in the two-level version.

The pixels p_i of an image $I(x_i, y_i)$ are labeled using LBP by thresholding each pixel of a 3×3 neighborhood with the value of the central pixel and treating the result as a binary number [84],[85],[86]. An example of LBP calculation is shown in Fig.3.8. The original 3×3 neighborhood is shown in Fig.3.8a. The following are the steps for determining the LBP measure:

- 1- By comparing the gray level values of neighbors with the central pixel value, the neighbors' labels are obtained using the binary code [0, 1] and it could be gained using equation (3.16). This binary code could be thought of as a binary pattern. It is labeled 1 if the examined gray level value of the pixel is greater than the gray level value of the central pixel, otherwise, it is labeled 0 [86], as illustrated in Fig.3.8b.

$$d'_i = \begin{cases} 0 & \text{if } I(x_i, y_i) < I(x_0, y_0) \\ 1 & \text{otherwise} \end{cases} \quad (3.16)$$

d'_i is the obtained binary code, d_i original pixel value at position i , and d_0 is the central pixel value.

- 2- Give weight to the corresponding pixels using the value 2^{i-1} , as shown in Fig.3.8c,
- 3- Multiplying the elements of Fig.3.8b by their respective weight kernel (Fig.3.8c) to obtain the outcome of this step as given in Fig.3.8d,
- 4- Finally, using equation (3.17), the LBP measure is calculated by adding the values of the eight pixels[86], as shown in Fig.3.8e.

$$I_{LBP} = \sum_{i=1}^8 d'_i 2^{i-1} \quad (3.17)$$

- 5- The central pixel is substituted by the obtained value (I_{LBP}). Each pixel and its 3x3 neighbors in the original image are processed to create a new LBP image.

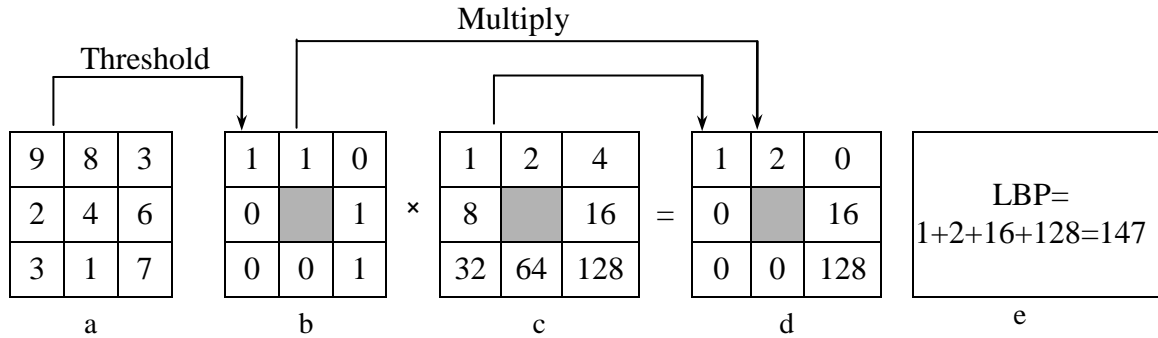


Fig.3.8 The stages of two-level version (LBP) texture unit calculation

The obtained LBP measure in this example is = 147.

The frequency of occurrences of certain binary patterns throughout the image is represented by a histogram[76]. The texture descriptor for the region is represented by the histogram of these LBP measures[86].

- ii) **Grey Level Co-occurrence Matrix-based Features (GLCM):** Also, it is known as a co-occurrence distribution. It represents and illustrates the spatial distance and angular spatial interrelation across a given image sub-region. In other words, it represents the number of occurrences of various gray-level combinations for each pixel and its specified neighbors in the image. The GLCM computes how often a pixel with a gray-level value i (grayscale intensity) appears horizontally, vertically, or diagonally when compared to neighboring pixels that have the gray level value j [87], [88].

The Grey Level Co-occurrence Matrix is a matrix with the same number of rows and columns as the image's gray levels (G). The Matrix's size represents the set of possible values. GLCM is computed based on two main parameters:

- The relative distance (d) between the pixel pair (distance or count to the next adjacent neighbor)
- The rotational angle (θ) or relative orientation, e.g. 0° , 45° , 90° , 135° , (8 orientations of adjacency), as seen in Fig.3.9.

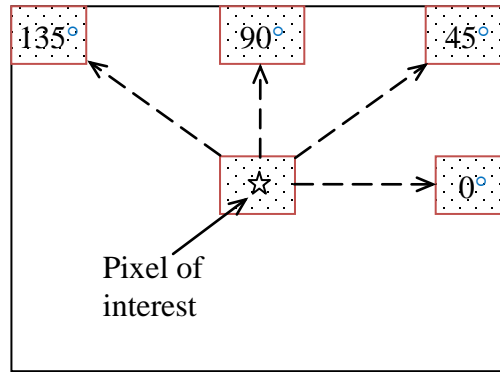


Fig.3.9 Directions/orientations (θ) of Adjacency in GLCM

In order to create a GLCM matrix, we must go through five steps[13]:

1. Generate a framework matrix,
2. Determine the spatial relationship between the reference and adjacent pixels,
3. Fill and complete the framework matrix by counting the occurrences,
4. Add the matrix to its transpose to make it symmetrical,
5. Normalize the matrix to convert it to probabilities

By applying some mathematical operations to the created GLCM values, different characteristics of the pixel distribution can be obtained. These types of operations were labeled as descriptors. Each descriptor is associated with a distinct texture characteristic. The following are some of the most often used descriptors (statistic parameters) that could be calculated from GLCM matrix[88]:

- **Maximum probability:** this metric determines the GLCM's strongest response. The possible value range is $[0, 1]$.
- **Correlation:** a metric that indicates how closely a pixel is connected to its neighbors throughout the entire image. The values range from 1 to -1, which corresponds to perfect positive and negative correlation contrasts, respectively.
- **Uniformity**, also known as energy, is a metric measure of consistency in the range $[0, 1]$. Uniformity is equal to one for a constant image.

- **Homogeneity:** determines how near the distribution of elements in the GLCM to the diagonal is in terms of spatial proximity. The value range is $[0, 1]$, with the highest value is obtained if GLCM is a diagonal matrix.
- **Entropy:** The randomness of GLCM's elements is being measured by entropy. If the image is not texturally uniform, then most elements of the GLCM have small values, indicating that the entropy is quite high.

The information about an image texture is described by these statistical parameters (Harrallic descriptors) and expressed in a vector form.

- iii) **Elements of GLCM:** The elements of GLCM have been converted to a vector (histogram), which appears the most successful approach in our work and has been adopted here. The GLCM matrix was chosen to be 8x8 in size (eight classes of grey level). We utilized a 1-pixel offset oriented to the east direction (from left to right). After normalization, GLCM elements (number of occurrences) turn into probabilities. The GLCM elements' probabilities of occurrence are transformed into a 64-bit vector, which appears in a histogram type of 64 bins. The GLCM-based histograms are shown in the fourth column of Fig.3.7.

The illustrations are made for different circumstances in the traffic scenario, which shows the candidate vehicle tracking windows in a sequence of frames. Three features are obviously not equally sensitive to the disruptions that occur during this phase. In the presence of shadow and partial occlusion, the edge feature histograms are fairly stable (appropriate histograms are of recognizable bipolar nature). Color histograms are the most susceptible to these impacts, whereas GLCM histograms are in the middle. None of the histograms are useful in the situation of complete blockage (final row) [15].

3.4 State estimation

Each new frame's measurement includes the calculation of the specified picture feature for each particle. The suitable histogram represents the contents of the considered particle (window) regardless of whatever single feature is chosen. The similarity metric is proportional to the particle's weight. In our approach, we utilized the Bhattacharyya distance (BD) between the actual particle (window) histogram and the reference histogram as information regarding local similarity at a certain position. The Bhattacharyya distance can be computed as follows:

$$BD^i(k) = \sqrt{1 - \sum_{u=1}^m \sqrt{hist_u^i(k) \cdot hist_u^{ref}(k-1)}} \quad (3.18)$$

where: i – particle index, k – frame index, u – bin index, m – total number of bins, $hist_u^i$ and $hist_u^{ref}$ refer to the extracted histograms of a considered particle's window and the reference one respectively.

The reference histogram $hist_u^{ref}$ is obtained using the initial one $hist(0)$, which is characterizing the initial window around the object, and the previous one $hist(k-1)$, which is characterizing the window around the last estimated position of the object.

The reference histogram $hist_u^{ref}$ is gained by utilizing the initial one $hist(0)$, which describes and represents the initial window around the object, and the previous one $hist(k-1)$, which describes the window surrounding the last estimated position of the tracked object. The equation (3.19) below represents the mathematical expression that is used for reference histogram calculation.

$$hist^{ref}(k) = \alpha \cdot hist(0) + (1 - \alpha) \cdot hist(k-1) \quad (3.19)$$

$\alpha \in [0,1]$ - Histogram weighting coefficient.

The particle's weight is computed using an exponential function, as shown in the equation (3.20) with $\sigma = 1.5$ as the deviation.

$$w^i(k) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(BD^i)^2}{2\sigma^2}} \quad (3.20)$$

The weights are then normalized as follow:

$$\bar{w}^i(k) = \frac{w^i(k)}{\sum_{j=1}^N w^j(k)} \quad (3.21)$$

The center (position) of the particle's window that is the most identical to the reference template (with the highest value of weight $w^i(k)$), is considered as the new estimate of state (position of the target).

$$(x_k, y_k) = (x_{w_{\max}}^i, y_{w_{\max}}^i), \quad (3.22)$$

$$w_{\max}(k) = \max(w^i(k)) \quad \text{for } i = 1, 2, \dots, N$$

or as a weighted sum:

$$(x_k, y_k) = \sum_{i=1}^N \bar{w}^i(k) \cdot (x^i, y^i) \quad (3.23)$$

3.5 Resampling

Resampling process is a significant step in particle filtering. It introduces the required feedback information from the measurements. Resampling step provides a way of using the likelihood of each particle \mathbf{x}_i to create a new set of particles in time \mathbf{k} for the next iteration in the tracking process[89], [90]. Each particle is chosen from the previous set of particles with a probability proportional to its weight, which implies that if the particles' weights are extremely low, they are removed with a high likelihood, and substituted with a duplicate of particles with large weights [72],[91]. In PF literature, resampling step has the particle degeneracy problem, which means that the variety of particles is reduced because particles with high significant weights are selected multiple times and particles with low weights are discarded[60], [90], [91], [92]. And the explanation for this is, when a few particles become dominant due to their high similarity measurements, the efficient number of particles is greatly lowered, the more of this takes place, the less particles there are that participate in the distribution's approximation. This has an impact on the tracker's capacity to respond appropriately to stochastic changes in the motion of the object. The effective number N_{eff} of samples is a signal of the degree of particles depletion; therefore, a resampling is carried out only when necessary instead of executing the resampling step at every iteration[60]. The PF's resampling condition is defined by a threshold, when N_{eff} decline below a nominated threshold, resampling begins. The effective number of particles can be determined using the formula (3.24).

A variety of techniques are available for resampling. The most popular resampling approach is select with replacement[60], [90], [92].

$$N_{eff} = \frac{1}{\sum_{i=1}^N (w^i)^2} \quad (3.24)$$

Among the several available resampling algorithms [92], we have chosen the most popular resampling approach, which is known as selective with replacement. The algorithm for this approach is as follows:

- (i) Calculate the cumulative weight wc^i (Cumulative sum of weights (CSW)) based on normalized particles' weights \bar{w} :

$$wc^i = \sum_{s=1}^i \bar{w}^s \quad (3.25)$$

- (ii) Generate a random number with a uniform distribution, $u_j \sim U[0,1)$

- (iii) **FOR** $j = 1:N$, From the initial set of particles, choose the particle x^i for replication based on following the condition:

If $u_j \in [wc^{i-1}, wc^i)$, then assign samples $x^j = x^i$

END FOR j

Fig.3.10 shows the cumulative sum of weights (CSW) graph calculated by wc^i

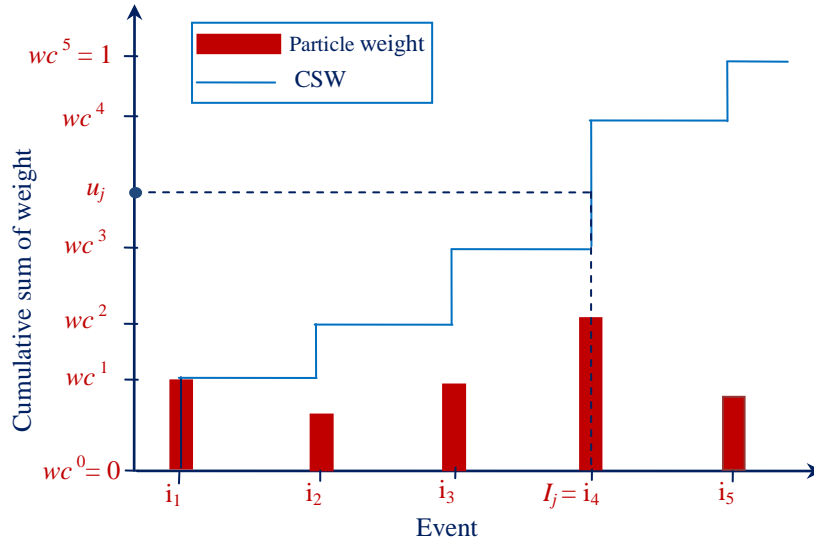


Fig.3.10 Cumulative weight graph: particle's weight (brown), the CSW (solid) and a transformation from standard uniform number to the event (dashed) [92].

Some particles with lower weights will be removed, while some particles with higher weights will be multiplied as a result of resampling[15].

3.6 Similarity measure

The similarity measure is an important step and plays a significant turn in the tracking process. It compares the target feature vector (histogram) with the feature vector of reference. In fact, it calculates the distance between the histograms of the reference window and the target window[41]. In this work, the Bhattacharyya distance has been used for similarity measures and assessment. The used Bhattacharyya distance expression is given in the equation (3.18).

For the proposed features used in this thesis, the histograms were calculated for RGB color with 21 bins for each color channel (63 bins, in sequence, together), 64 bins for an edge and 64 bins for texture.

The similarity measure criteria utilized for the evaluation between the two histograms are based on Bhattacharyya distance as given in the equation (3.26):

$$sim^i(k) = 1 - BD^i(k) \quad (3.26)$$

The predicted position of the moving target is chosen as the position of the particle where the similarity is the best. Fig.3.7 showed typical histograms for RGB color, edge gradient orientation, and texture of the reference target window (#310) and candidate window (#320, #340, and #360).

3.7 Multi-Feature-Based Tracking Algorithm

Many of the particle filters mentioned in the literature rely on a *Single-Feature-based Particle Filter* (SFPPF) for tracking vehicles in the video sequence, but they are suffering from many individual drawbacks, and have weakness ambiguities because a single feature in some circumstances doesn't supply enough information and reliable performance, especially at the existence of visibility noise and typical disturbances[15], [43] (e.g. the influence of luminance changes, shadows, and texture ambiguity during object's rotation). Accordingly, tracking using several features can offer more information and a more accurate description of the object, as well as increase the tracking system's robustness.

The image features mentioned in sections 3.3.1, 3.3.2, and 3.3.3 would be the candidate features to be used for SFPPF. The primary idea beyond this Multi-Feature-based Particle Filter (MFPF) was that by fusing the results of various SFPPFs, one might overcome the shortcomings of each SFPPF. The relative significance of all SFPPF outputs will be adaptively modified based on the assessed quality of these outputs in order to give priority to picture features that are less influenced by the present disturbance situation[15].

There are certain additional steps and parameters that should be mentioned in the construction of the MFPF algorithm in comparison to the proposed original SFPPF steps stated in Section 3.1.3:

- Initial distribution of particles – made as the same one used for all three single features (SF). Assuming the dispersion radius is DR_0 and due to the significant initial uncertainty of the object's velocity, the initial dispersion radius, has been doubled ($DR_0 = 2 \cdot DR$), while the radius utilized for the following steps is DR_0 ,
- The average similarity AS_n of all features (SF_n), is determined using the average value of Bhattacharyya distances BD^i of all particles. This is a fundamental parameter that designates the relative significance of a feature,
- The similarity metric is determined by the equation (3.26),
- For any SF_n , the level of confidence LC_n is defined as the ratio of real AS_n to initial $AS_n(1)$ in the first frame following initialization. This level of confidence could be more than one (in the case of successive average similarity AS_n is greater than $AS_n(1)$).
- After normalizing the three LC_n values, the relative value (weighting factor, WF_n) for each SF_n is calculated.

$$WF_n(k) = \frac{LC_n(k)}{\sum_{j=1}^3 LC_j(k)} \quad (3.27)$$

Equation (3.28) is used to estimate the final state by the weighting factor and position of the object with maximal weight

$$(x_{obj}, y_{obj}) = \sum_{n=1}^3 WF_n \cdot (x_{w_{\max}}^n, y_{w_{\max}}^n) \quad (3.28)$$

- If the AS_n that is correlated to LC_n is less than the threshold value AS_n^{thr} , the LC_n is set to zero, and the SF_n is not considered in the final state estimation.
- If all AS_n are less than their specified threshold values (as in full occlusions), PF is switched to "prediction mode," to estimate the next new position relying on past predictions, rather than new measurements.
- During the full-occlusion phase, the radius of particles scatter (DR) is gradually expanding, while they stay at their nominal value ($DR = DR_0$) in the case of "partial occlusion or non-occlusion phase"
- The reference histograms (reference model) that will be used during the "prediction phase" would be the last reference histograms used on the previously estimated position in "estimation mode".
- Resampling has a slightly different meaning in the MFPP algorithm environment than it does in the SFPP algorithm environment. After all three features have been resampled; the final step is to extract particles "N" from all three sets of resampled particles at the same time.

Fig.3.11 below shows the overall flow chart for the proposed MFPP algorithm.

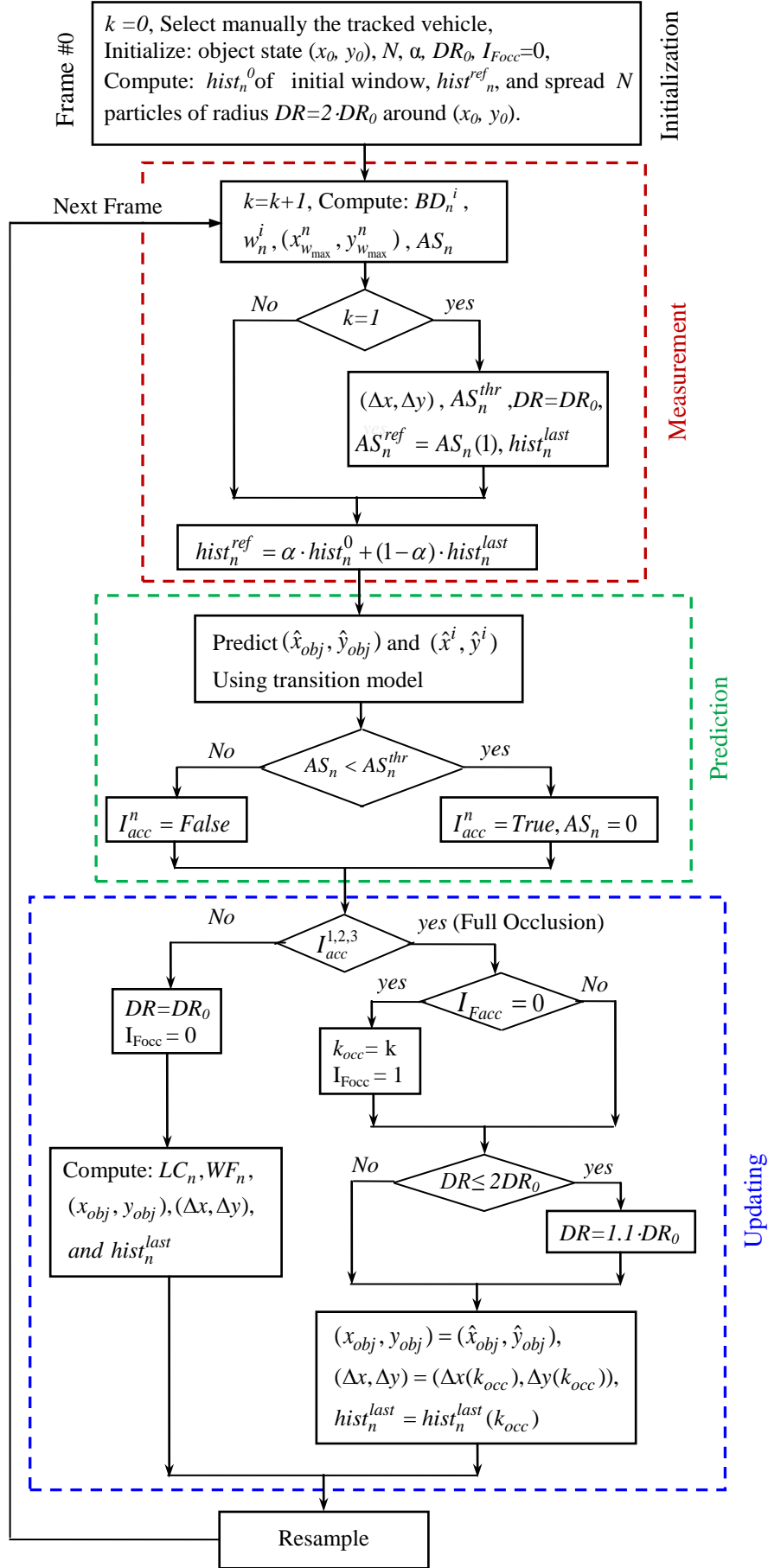


Fig.3.11 MFPPF flow chart algorithm

where:

- I_{occ}^n are the indicator for the appearance of occlusion in $SFPF^n$,
- k_{occ} is the time when the occlusion began,
- I_{Focc} is an indicator, indicates full occlusion,
- AS_n^{thr} are average similarities thresholds for each feature,
- $(x_{w_{max}}^n, y_{w_{max}}^n)$ are locations where the weights are largest for $SFPF^n$,
- $hist_n^0$ represent the initial reference histograms of $SFPF^n$,
- $hist_n^{ref}$ are $SFPF^n$'s reference histograms for features that have been updated,
- $hist_n^{last}$ are the last-updated histograms,
- (x_{obj}, y_{obj}) the obtained estimate of position for the object,
- $(\hat{x}_{obj}, \hat{y}_{obj})$ is the predicted position of the object,
- (\hat{x}^i, \hat{y}^i) are the predicted positions of the particles.

The MFPPF algorithm steps for integrated Features for the flowchart given in Fig.3.11 are as follows:

(i) **INITIALIZATION:** at $k = 0$ (k is frame index),

- Initialize the values: $[h_x, h_y]$, N , DR_0 , I_{Focc} , α
 - Where $[h_x, h_y]$, is the dimension of window, N is number of particles, DR_0 is distribution radius, I_{Focc} is an indicator, indicates complete occlusion occurred, and α , is forgetting factor.
 - Manually choose the monitored object at its initial location (x_0, y_0) ,
 - Set the rectangular window of dimension $[h_x, h_y]$ around object's centroid,
 - Compute initial histograms for initial window:
 - $hist_1^0 = hist_{RGB}^0$, initial RGB histograms
 - $hist_2^0 = hist_{HOG}^0$, initial edge gradient histograms
 - $hist_3^0 = hist_{GLCM}^0$, initial gray level co-occurrence matrix histograms.
 - Set reference histograms to initial ones: **For** $n=1:3$, $hist_n^{ref} = hist_n^0$, **end for**
 - Create and spread a set of N particles around the initial position (x_0, y_0) with a radius of $2 \cdot DR_0$ according to adopted distribution.
-

(ii) **MEASUREMENT STEP:** at $k = k + 1$,

- **For** $n = 1:3$,
 - **For** $i = 1:N$,
 - $BD^i(k) = \sqrt{1 - \sum_{u=1}^m \sqrt{hist_u^i(k) \cdot hist_u^{ref}(k-1)}}$, Calculate Bhattacharyya distances for each particle (Equation (3.18))
 where, k is frame number, u is index of bin, m is number of bins, $hist_u^i$ is particle window's histogram and $hist_u^{ref}$ is the reference histogram
 - $sim^i(k) = 1 - BD^i(k)$, Calculate the similarity measures for all N particles (equation (3.26)),
 - $w^i(k) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(BD^i)^2}{2\sigma^2}}$, Calculate the particles' weights ($\sigma = 1.5$) (Equation (3.20)),
 - $\bar{w}^i(k) = \frac{w^i(k)}{\sum_{j=1}^N w^j(k)}$, Normalize the weight (Equation (3.21)),
 - **End for** i
 - $\bar{w}^n(k) @ (x_{w_{max}}^n, y_{w_{max}}^n) = \max(\bar{w}^i)$, Find positions $(x_{w_{max}}^n, y_{w_{max}}^n)$ of particles with largest weights for each feature
 - $(D_x^n, D_y^n) = \left[(x_{w_{max}}^n - x_{obj}), (y_{w_{max}}^n - y_{obj}) \right]$, Calculate increments for each feature
 - $AS_n = \frac{1}{N} \sum_{j=1}^N sim_j$, Calculate average similarities,
- **End For** n
- **if** $k = 1$,
 - $(\Delta_x, \Delta_y) = \frac{1}{3} \sum_{n=1}^3 (D_x^n, D_y^n)$, Initial calculation of prediction increment,
 - **For** $n = 1:3$,
 - $AS_n^{ref} = AS_n(1)$, define reference similarities,
 - $AS_n^{thr} = 80\% \text{ of } AS(1)$, set average similarities thresholds
 - $hist_n^{last} = hist_n @ [(x_0 + \Delta_x), (y_0 + \Delta_y)]$, Initialization of the last histograms in the first step
 - **End For** n
- **End if**
 - $hist_n^{ref} = \alpha \cdot hist_n^0 + (1-\alpha) \cdot hist_n^{last}$, modifying reference histogram

(iii) PREDICTION STEP:

- predict the moving vehicle's state $(\hat{x}_{obj}, \hat{y}_{obj})$ based on state transition model (Equation(3.8))
 - **For** $n = 1:3$, specify the positions of particles (\hat{x}^i, \hat{y}^i) according to transition model,
 - **End For** n
 - **For** $n = 1:3$,
 if $AS_n < AS_n^{ref}$, check the feature validity
 $I_{occ}^n = True$,
 $AS_n = 0$,
 Else
 $I_{occ}^n = False$
 End if
 - **End For** n
-

(iv) UPDATING STEP:

- **if** $I_{occ}^1 \cdot I_{occ}^2 \cdot I_{occ}^3 = True$, full occlusion is occurred
 if $I_{Focc} = False$,
 $k_{occ} = k$, Remember the index of last frame before occlusion
 $I_{Focc} = 1$, switch occlusion flag to full occlusion,
 End if,
 if $DR \leq 2 \cdot DR_0$, Gradual increasing radius of spreading.
 $DR = 1.1 \cdot DR_0$
 End if,
 $(x_{obj}(k), y_{obj}(k)) = (\hat{x}_{obj}(k), \hat{y}_{obj}(k))$, new object state in full occlusion state
 $(\Delta_x(k), \Delta_y(k)) = (\Delta_x(k_{occ}), \Delta_y(k_{occ}))$, position increment in full occlusion state
 For $n = 1:3$, $hist_n^{last} = hist_n(k_{occ})$, **End For** n , remember the last valid histograms
 - **Else**, no complete occlusion
 $DR = DR_0$
 $I_{Focc} = 0$
 For $n = 1:3$, $LC_n = \frac{AS_n}{AS_n^{ref}}$, **End For**
-

For $n = 1:3$, $WF_n = \frac{LC_n}{\sum_{j=1}^3 LC_j}$, **End For**

For $n = 1:3$, $(\Delta_x(k), \Delta_y(k)) = \sum_{n=1}^3 WF_n \cdot (D_x^n, D_y^n)$, **End For**

$(x_{obj}(k), y_{obj}(k)) = [x_{obj}(k-1), y_{obj}(k-1)] + (\Delta_x(k), \Delta_y(k))$, Calculate estimated vehicle's position,

For $n = 1:3$,

$hist_n^{last} = hist_n @ [x_{obj}(k), y_{obj}(k)]$, Calculate histograms on updated position,

End For

• **End if**,

(v) Resampling step:

Resample the particles in all three features by excluding and substituting any particle that fulfills the removal requirement, according to section 3.5.

(vi) GO TO step 2 (To start new iteration).

Chapter 4

The Performance Evaluation of the Proposed SFPF Algorithm

4 The performance evaluation of the proposed SFPP algorithm

4.1 Testing the environment for object tracking using different features based Particle Filter

One of the most important researches in the visual tracking field is the development of systems that automatically analyze or monitor traffic activities. Tracking of targets has its own challenges when the targets pass through many disturbances such as movement of the vehicles close to each other, merging, appearing of shadows, or maneuvering, etc. Often the goal of tracking is to obtain a record of the correct trajectory of single or multiple targets over time and space, by processing the available information from the distributed sensors or any other means. For that purpose, it is required to attain accurate and dependable vehicle tracking in traffic images. So, to evaluate the performance of the proposed algorithm mentioned in the previous chapters, two steps have been taken. In the first step, synthetic traffic scenarios have been created to simulate moving vehicles in two directions on a round road of two lanes, using MATLAB [93]. The vehicles have been assumed to move at a constant velocity. This program has been generated for the purpose of designing and testing a moving vehicle tracking algorithm. Fig. 4.1 represents a top view of the suggested traffic scenario. The second step is the implementation and testing of the proposed tracking algorithm on the created synthetic scenarios [94]. This algorithm can be utilized and performed in a real video sequence.

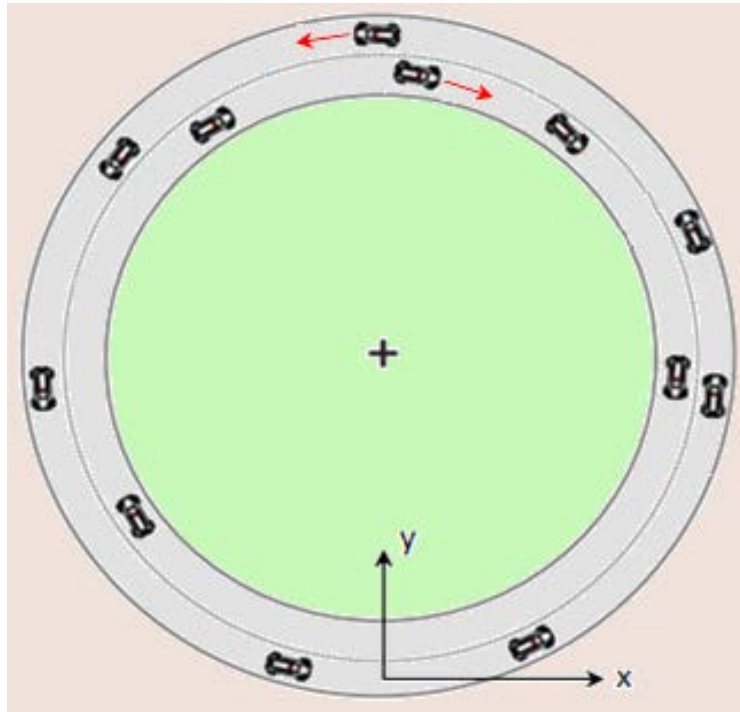


Fig. 4.1 Top view of the traffic scenario [93]

4.2 Testing the proposed tracking algorithm SFPF using multiple descriptors of the image independently

In this section, we are going to test the proposed algorithm (SFPF algorithm) of object tracking based on three individual and independent features (image descriptors) by means of presenting two cases. The first one is a single object tracking in a created artificial video sequences for three different scenarios; the second one is single object tracking in real video sequences for two different scenarios.

In general, any traffic scenario might be exposed to many different environmental conditions. The testing scenarios illustrated in the following sections have been used to achieve the following targets:

- i) For testing the algorithm on created artificial and real video scenarios for the case of a single movable object. For that purpose, we have tested some disturbances' effects on the tracking process. Typical situations to be tested were the cases: without disturbances, disturbances caused by shadows, maneuvering, and partial occlusion,
- ii) In order to present a comparative analysis of the application of the object tracking algorithm (SFPF) using three independent features (image descriptors).

For the proposed algorithm (SFPF) we have adopted independently the RGB color space, edge intensity, and texture as separate descriptors to be used in SFPF algorithm.

4.2.1 Synthetic Traffic scenarios generation for testing of the tracking algorithm

Traffic simulation has been an often-used tool in the study of traffic systems. The synthetic traffic scenarios have been created on a suggested circular two-lane road as shown in Fig. 4.1 and it is implemented via Matlab program that simulates the motion of string vehicles in both directions on the road lanes [93].

The tracking method for video data and information that we use is based on a window-matching manner, which involves determining the grade level of likeness between regions in sequential images. In this manner, an object can be recognized, and its position deduced in the later frames [95]. For the synthetic video sequence, relatively small windows of size equal to 32×32 pixels are used for window matching purposes with the criterion to maximize the proper similarity function, as it was used in [43].

The assumed ego-car in the synthetic proposed scenarios is the red one, and the parameters for the synthetic video sequence simulation are:

- The velocity of the target vehicle in all cases is 90 km/h.
- The time interval between frames is 50 ms.
- The number of particles used in PF was 150 in all cases.

- The spreading radius of the particles was 10 pixels in Cases-I and II, and 15 pixels in Case-III.
- The framework window size is 32×32 pixels.
- 1 meter \approx 7 pixels.

There are three proposed scenarios of moving vehicles (for the synthetic sequence of images) used for testing the tracking algorithm:

1) Case-I: Tracking the non-maneuvering target car without disturbance

This was the case used for the basic verification of the tracking algorithm in situations where there are no disturbances and no obstacles on the road, as shown in Fig. 4.2:

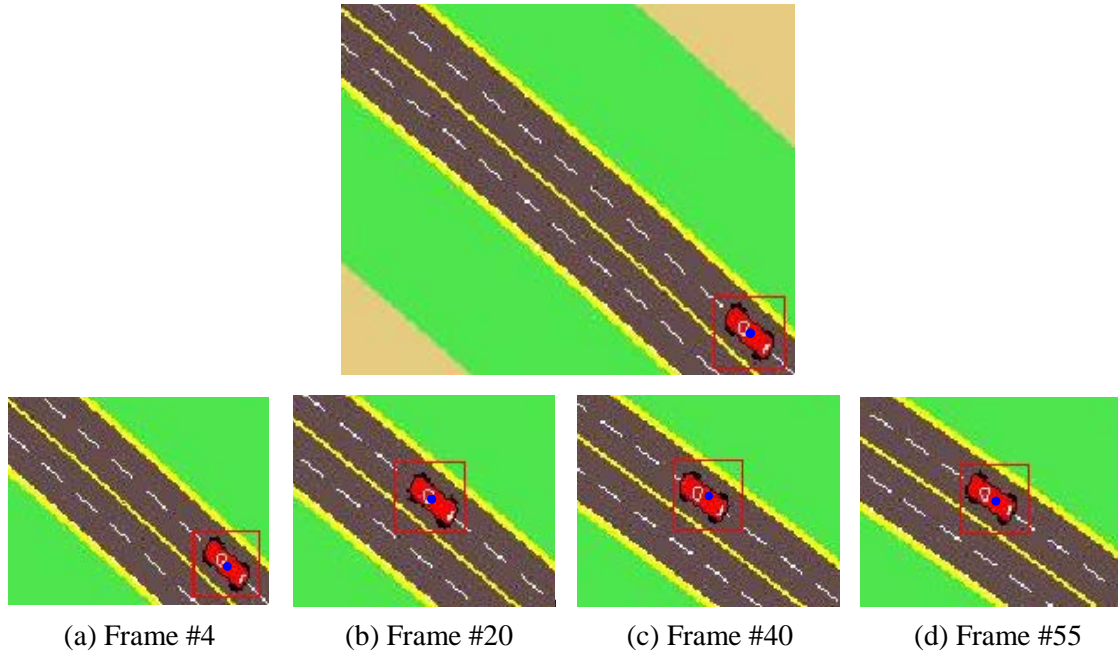
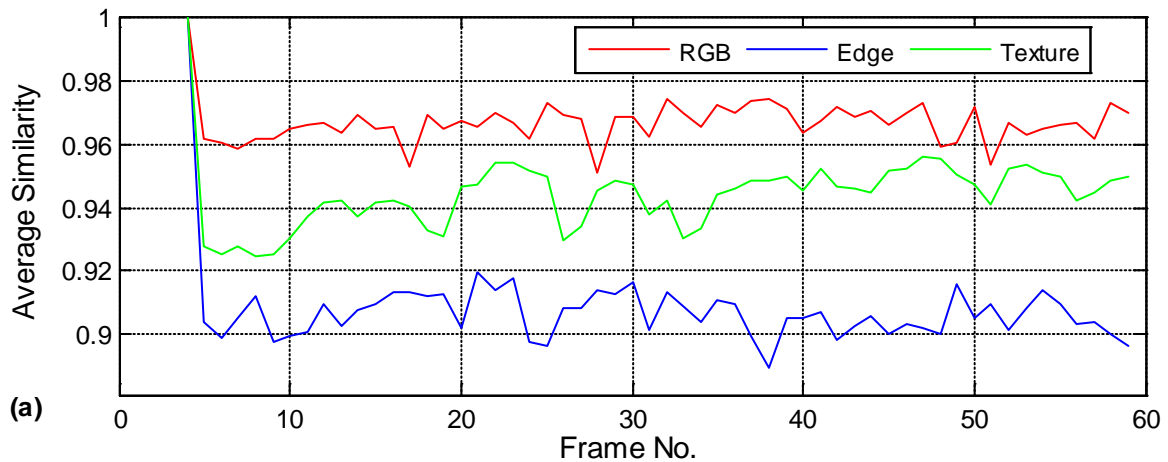


Fig. 4.2 Tracking of the vehicle without disturbances on the road

Fig. 4.3 below shows the average similarity (for all particles) and the deviation between the actual vehicle position and the estimated (updated) target positions respectively, using the three features independently.



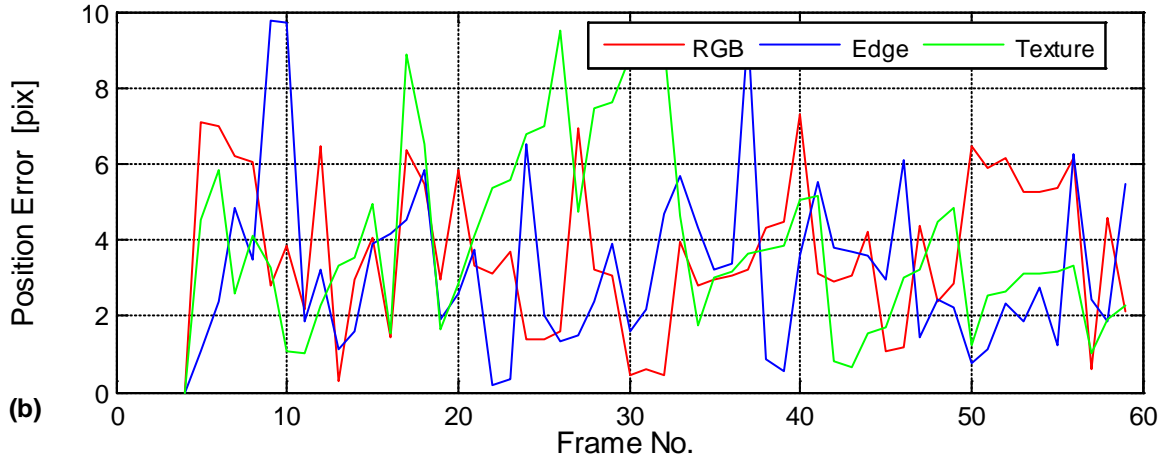


Fig. 4.3 Case-I a) Average similarities, b) Tracking errors

From Fig. 4.3 we can see that the errors are approximately the same for all three descriptors (root mean square errors are 4.18, 3.94, and 4.61 pixels, respectively for color, edge, and texture). Average similarity slightly gives preference to the color descriptor.

2) Case-II: Tracking a target car with disturbances in the image (shadow produced by the ego-car itself and shadows produced by the road environment)

In this scenario, the disturbances such as the ego-car shadow and environment's shadows such as those produced by clouds, trees, buildings, or by any other things besides the road, that appear abruptly at different intervals have been inserted. Fig. 4.4.b illustrates the target car with shadow produced by the car itself ($k = 15$), and Fig. 4.4.c shows the target car when it is affected by a building's shadow beside the road ($k = 45$).

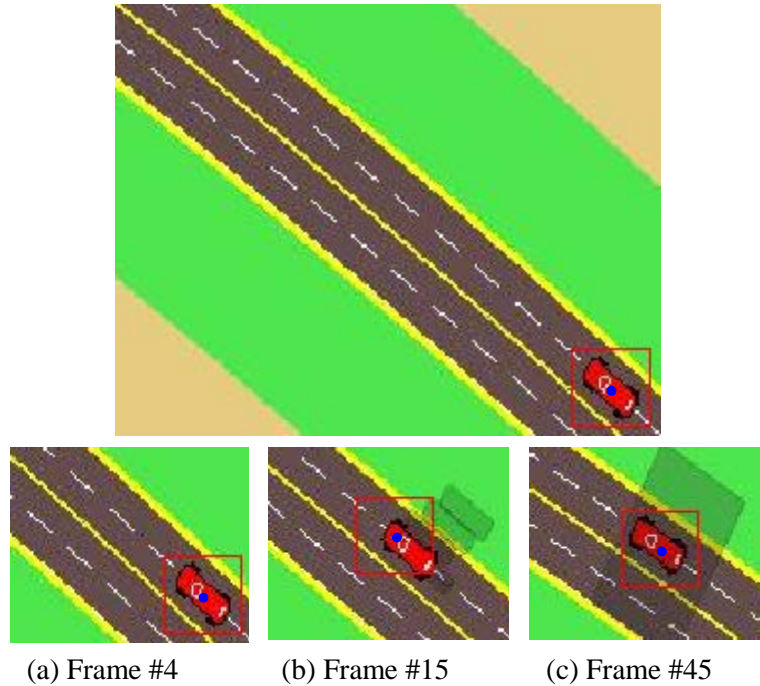


Fig. 4.4 a) Reference frame (#4), b) Target car with its shadow ($k=15$), c) Tracked car within the building's shadow ($k=45$)

The obtained average similarity and deviation between the actual and estimated position of the target vehicle can be seen in Fig. 4.5 below.

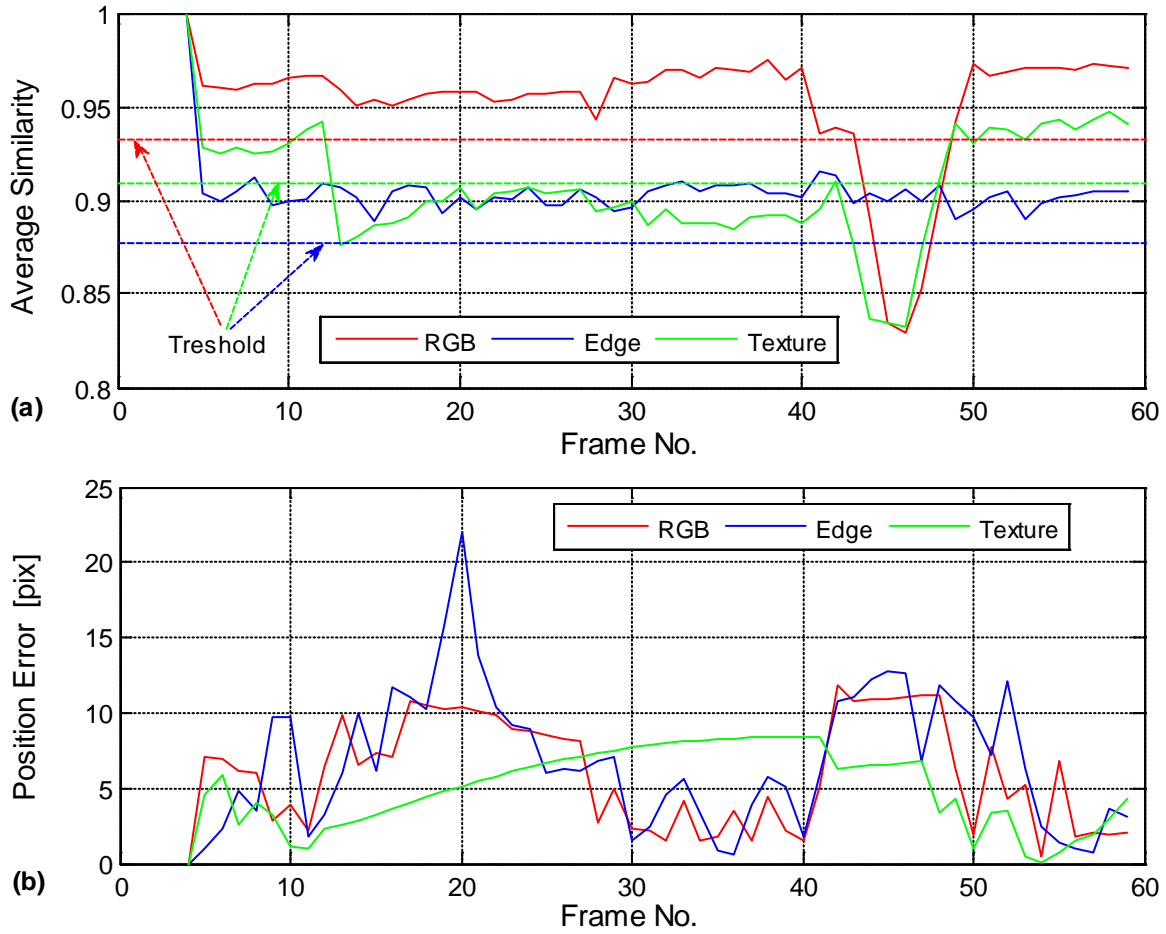


Fig. 4.5 Case-II a) Average similarities, b) Tracking errors

The error diagram shows that the color and texture descriptors are more sensitive to the shadow in both cases. The average similarity of the color and texture descriptors is dropped to 0.83 at the building's shadow interval, and the root mean square errors are 6.94, 8.19, and 5.61 pixels for color, edge, and texture respectively (giving preference to edge descriptor).

3) Case-III: Tracking a maneuvering target car without disturbance

Fig. 4.6 illustrates the situation of tracking stages of car maneuver bypassing another car, before over-taking ($k = 25$), the two cars in parallel ($k = 39$), and after over-taking ($k = 55$).

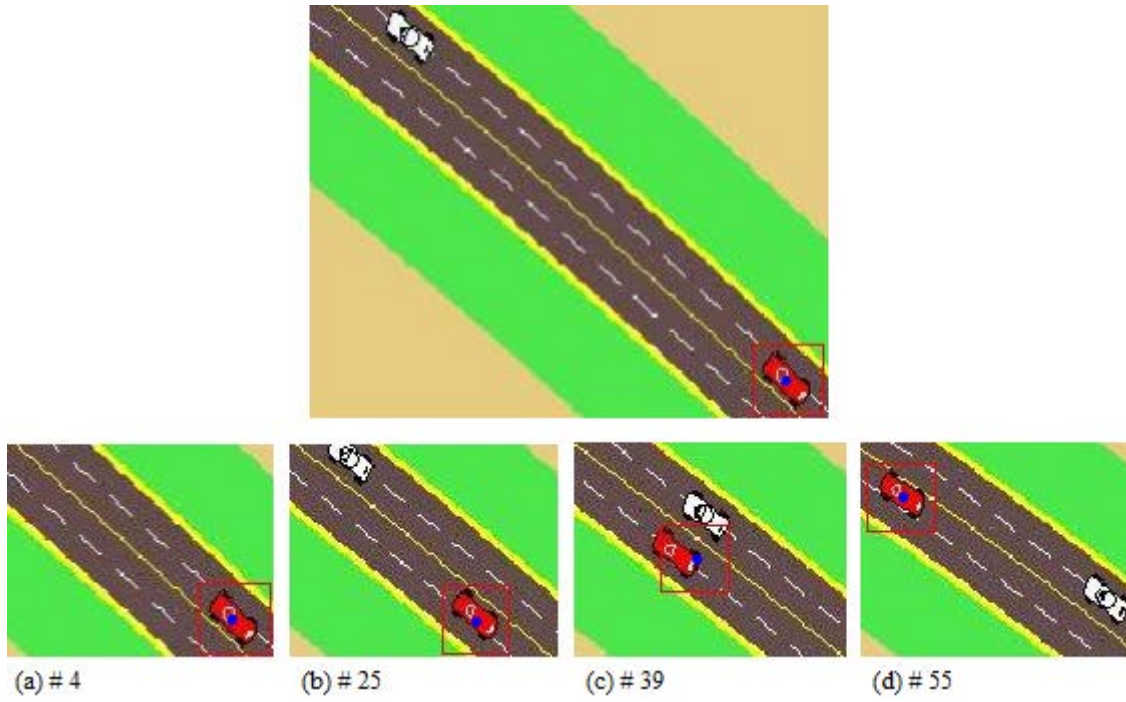


Fig. 4.6 Different status intervals frames, a) Reference frame (#4), b) before over taking ($k=25$), c) 2-cars in parallel ($k=39$), d) after over taking ($k=55$)

The average similarity and the deviation between the actual and estimated positions respectively, for the target vehicle can be seen in Fig. 4.7. The maneuvering stage occurs between the 4th frame and 70th frame.

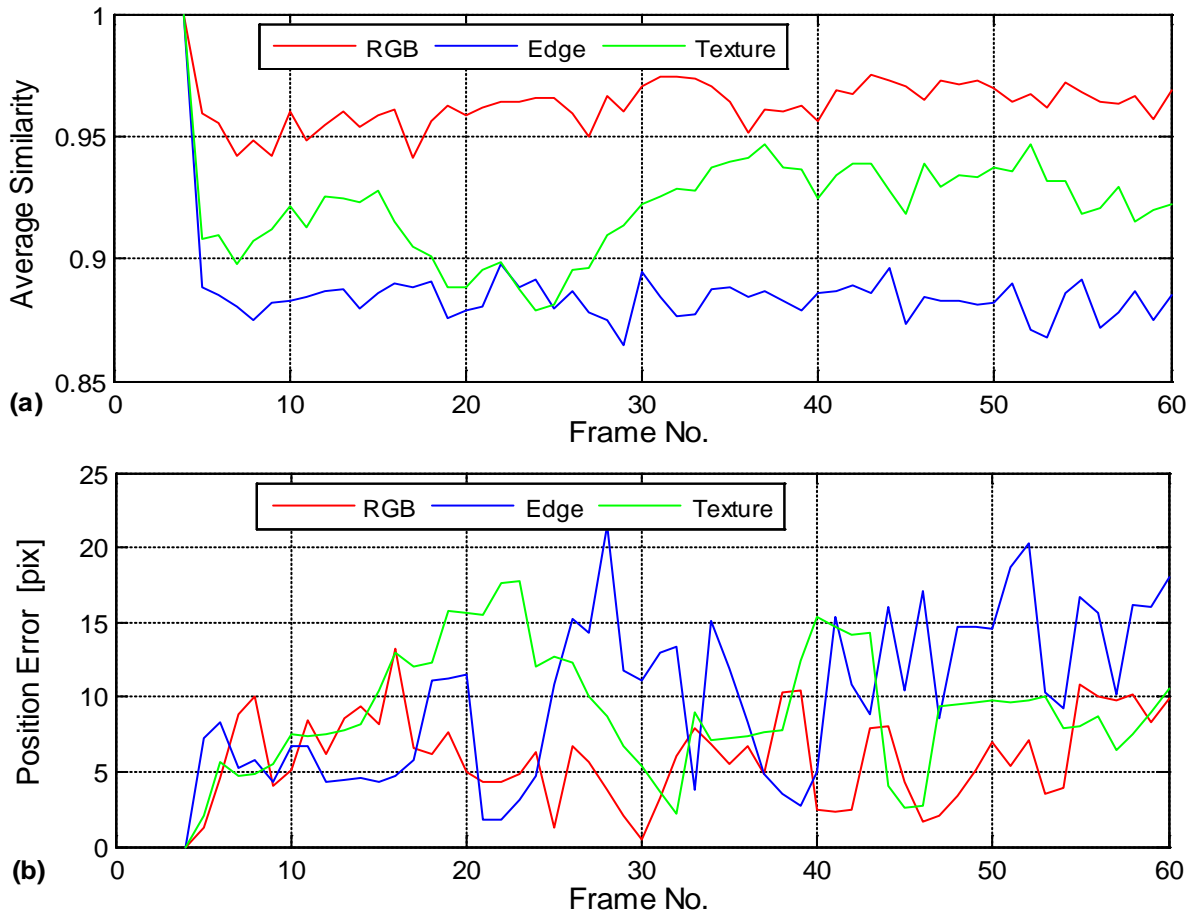


Fig. 4.7 Case III, a) Average similarities, b) Tracking SFPP errors

We can see from Fig. 4.7 that the color is the appropriate and favorite compared with other descriptors; and the root mean square error is 6.67, 11.26, and 9.92 pixels for color, edge, and texture respectively.

Table 4.I summarizes the RMS errors in pixels for the three preceding examples of artificial sequence cases using the SFPF algorithm. Looking at the results obtained from the synthetic sequences (cases 1 to 3), in Case I Edge offers the best RMS error, and in Case II Texture grants the best RMS error, while in Case III, Texture provides the best RMS error.

Table 4.I RMS tracking errors in pixels

	Case I	Case II	Case III
RGB	4.18	6.94	6.67
Edge	3.93	8.19	11.26
Texture	4.61	5.61	9.92

4.2.2 Single object tracking in real-video sequences using three independent image descriptors

The proposed tracking algorithm has been tested by using the created artificial video images in the previous section. In this section, we are applying and implementing the proposed algorithm in real-video sequences for tracking a single vehicle based on each feature individually using PF. Two cases of real-video sequences (Case-IV and Case-V) have been selected here. The first case is 'Tracking the target car in real video sequences in the presence of the car's shadow itself and trees' shadow as disturbances', and the second case is 'tracking of the maneuvering car in real-video sequences with partial and full occlusions'. The assumed simulation parameters for the selected real-video sequence are:

- The interval between frames - 10.
- The number of particles used in PF was 200 particles for both cases.
- The spreading radius of the particles was 20 pixels for both cases.
- The framework window size ($H \times w$) is 120×110 and 60×64 pixels for Cases-IV and V, respectively.
- 1 meter \approx 63 pixels for Case-IV and 1 meter \approx 16 pixels for Case-V.

1) Case-IV: Tracking the target car in a real video sequence with disturbances of car and tree shadow

This case describes the tracking situation in the existence of car shadow and tree shadow. Fig. 4.8 shows the overall scenario followed by the target windows and color histogram respectively for the reference frame ($k = 235$), car at tree's shadow frame ($k = 263$), and car after tree shadow frame ($k = 291$).



Fig. 4.8 Overall scenario and target vehicle window at different time

A typical histograms for the three features respectively: color, edge, and texture for target vehicle window at three different times (k), are shown in Table 4.II.

Table 4.II Histograms for color, edge, and texture descriptors

RGB			
Edge			
Texture			
Frame No.	#235	#265	#290

The average similarity and the deviation between the actual and estimated position of the target vehicle respectively are shown in Fig.4.9.

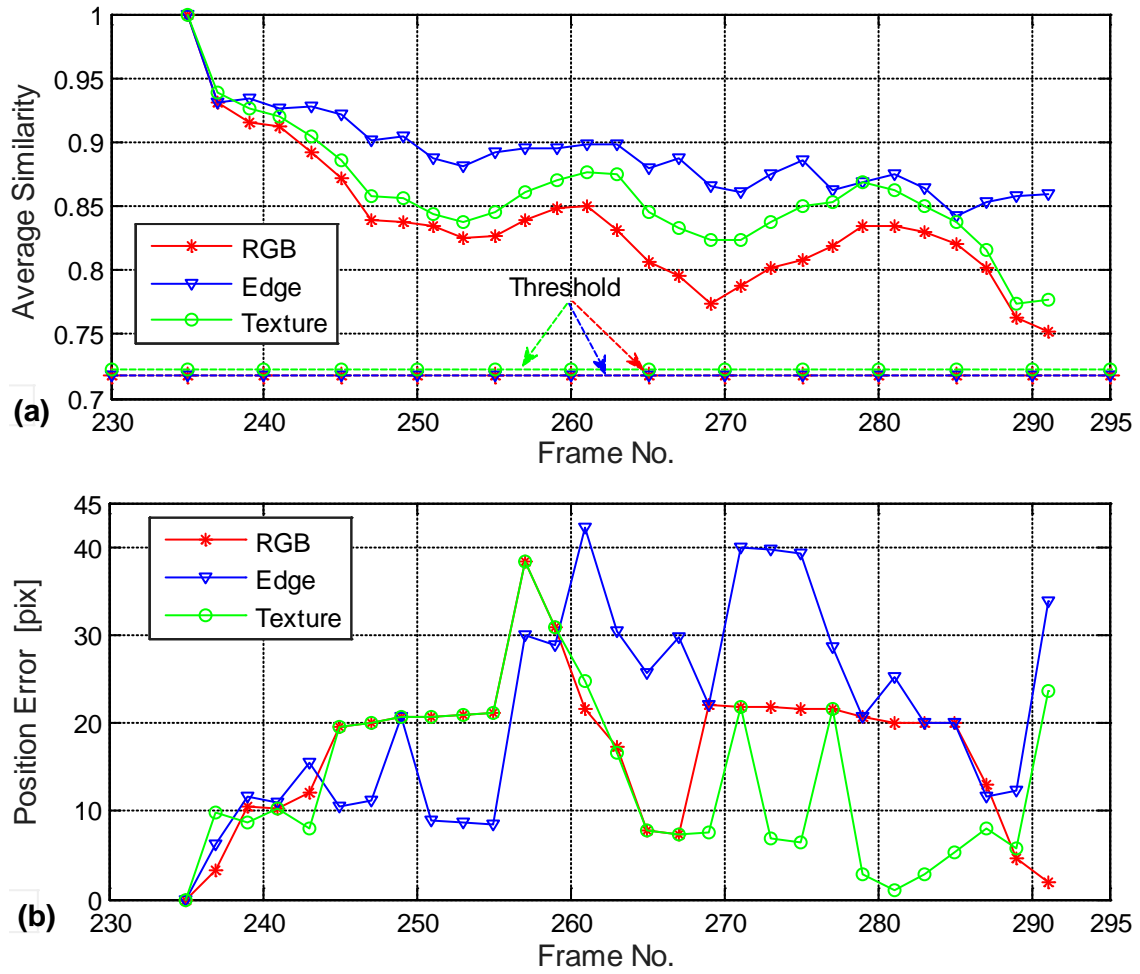


Fig.4.9 Case-IV (a) Individual average similarities, (b) SFPP errors

We can see from the average similarity in Fig.4.9.a that the edge is a favorable descriptor; while the mean square errors are 18.94, 24.01, and 16.69 pixels respectively for color, edge, and texture.

One should note that there are some differences in comparison to the synthetic sequence of this type:

- (i) Size of the frame – larger than before
- (ii) Variable shadow all the time
- (iii) Both types of shadows simultaneously
- (iv) The correlation between pixels and meters is variable and different (1 meter \approx 64 pixels).

2) Case-V: Tracking of maneuvering target car in real-video sequences with partial and full occlusion

This case describes the tracking situation of the target vehicle exposed to partial and full occlusions due to the tree beside the road. One could say that this case is the most complex scenario because of the following:

- (i) Decreasing the size of the car.
- (ii) Partial and even full occlusions.

(iii) Maneuver during the full occlusion phase.

Fig. 4.10 illustrates the overall scenario and some important frames (a, b, and c) that contain the target windows and corresponding color histograms representing these situations. The yellow and red dots inside the red windows represent the actual and estimated target positions respectively.

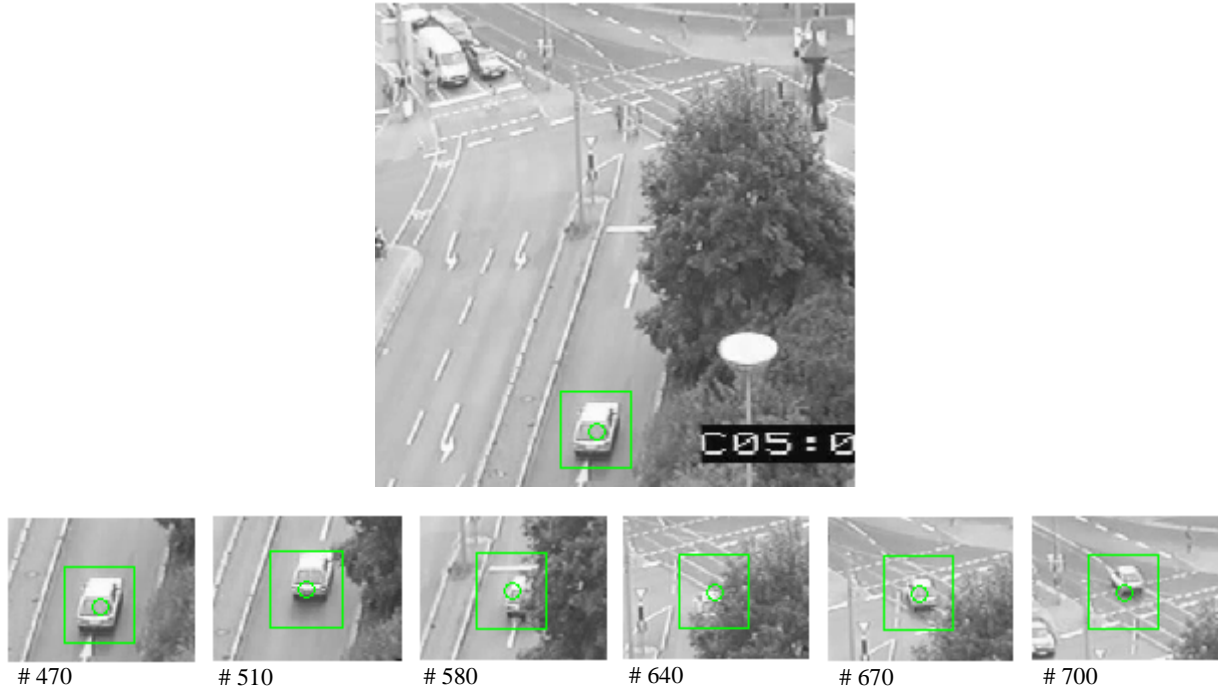


Fig. 4.10 Overall scenario and target vehicle window at different time

Table 4.III shows typical histograms for the three features respectively: color, edge, and texture of the target vehicle window at three different times.

Table 4.III Histograms for color, edge, and texture descriptors

RGB			
Edge			
Texture			
Frame No.	#470	#580	#640

Fig.4.11 below shows the average similarity and the deviation between the actual and the estimated (updated) target positions using the three descriptors independently respectively.

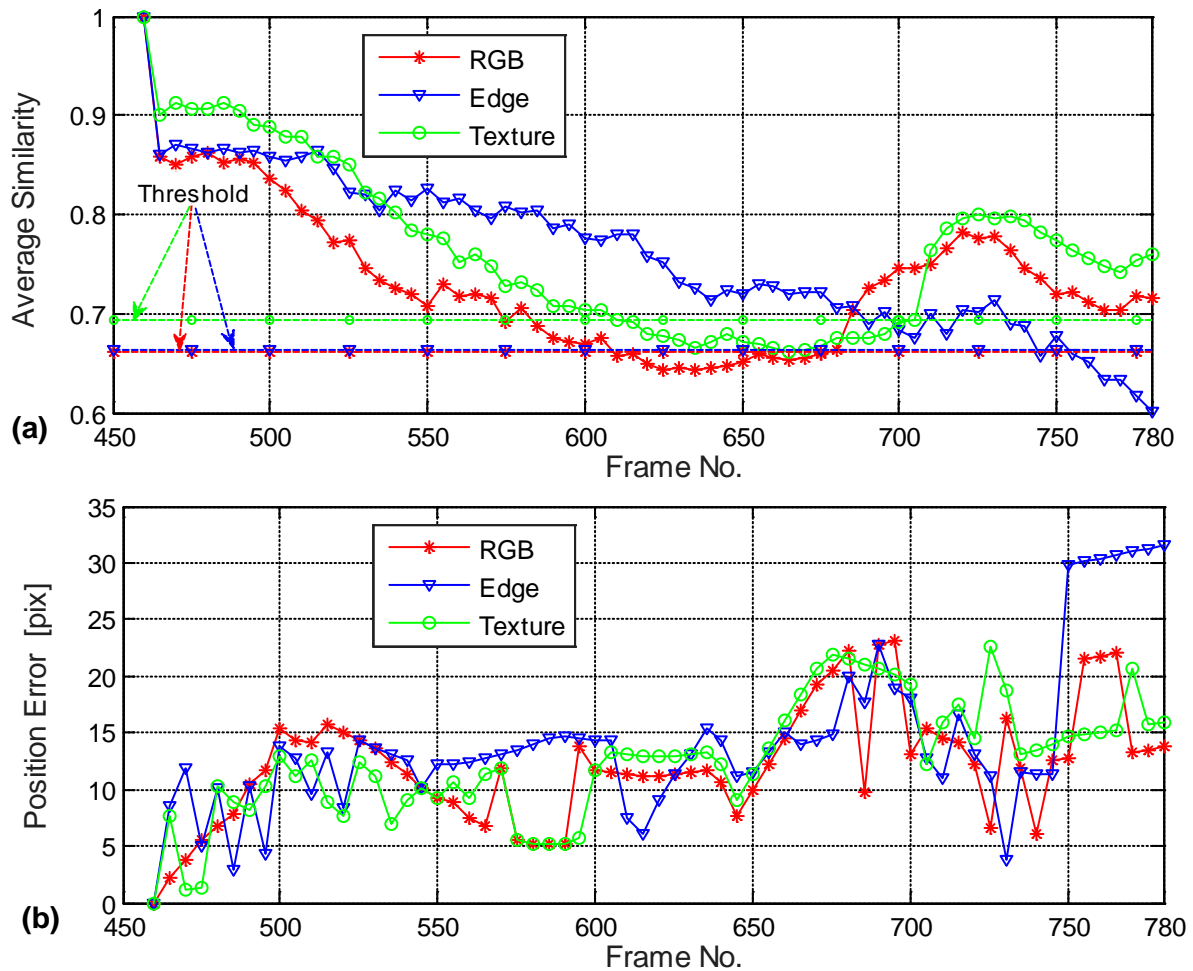


Fig. 4.11 Case-V: (a) Individual average similarities, (b) Individual SFPF errors

According to the average similarity Fig. 4.11.a, it's obvious that all descriptors mostly have comparatively near values. Error values in Fig. 4.11.b show that color and edge, as well as texture, have relatively similar values at different periods. The RMS errors for color, edge, and texture are 13.10, 15.77, and 13.50 pixels respectively.

Table 4.IV illustrates the obtained RMS errors in pixels that present the SFPF results for the two real scenarios (cases IV and V) for different attitudes and behavior.

Table 4.IV RMS TRACKING ERRORS (PIXELS)

	Case IV	Case V
RGB	18.94	13.10
Edge	24.01	15.77
Texture	16.69	13.50

From the studied, tested, and analyzed cases for a single object in synthetic video sequences and the real-video sequences using the proposed SFPPF algorithm to track a moving object, we can summarize the following:

- We have presented a comparative analysis using three individual features/image descriptors (independently) by the application of the proposed object tracking SFPPF algorithm.
- In the case of the target vehicle (Case-I), where it is without maneuvering and disturbances, the tracking errors are mostly the same for all three descriptors.
- In the case of tracking of the target car (Case-II) with its self shadow, and the shadow produced by the road environment, it's obvious that the color and texture descriptors are more sensitive in comparison to the edge descriptor, as shown in Fig. 4.5.b.
- The maneuvering of the undisturbed target vehicle (Case-III) shows that the edge descriptor is the most sensitive to rotation, as illustrated in Fig. 4.7, and the color cue gives the best accuracy compared to the others.
- In the case of tracking the target car (real video) with its shadow and tree's shadow (Case-IV), one can conclude that the edge is the favorable descriptor in the first half of the journey whereas texture is favorable in the second half of the journey, according to Fig.4.9.b.
- The vehicle in maneuver status (Case-V), is being the most complex scenario in the presence of the mentioned disturbances. Looking at the similarity results' figures, favorability in some periods has been given to the color, and at other times it is given to the texture, whereas at another time it is given to the edge.
- It could be said that according to different scenarios tested here, none of the analyzed descriptors can be declared as the best one in all cases. To overcome this ambiguity, some fusion of individual results obtained by the different descriptors should be made into one integral tracking algorithm based on the three analyzed descriptors, and this will be implemented and applied in the next chapter.

Chapter 5

Performance Evaluation of MFPP Tracking Algorithm in Comparison to SFPP Algorithm

5 Performance Evaluation of MFPP Tracking Algorithm in Comparison to SFPF Algorithm

Through describing and implementing the particle tracking techniques to track a moving vehicle in a video sequence in the earlier chapters using a Single-Feature Particle Filter "SFPF", we can say that none of the used descriptors can be suggested and declared as the best descriptor could be used alone as the comprehensive nominee in all different circumstances because these features have shown their own sensitivities in some conditions. The color-based features are usually sensitive to illuminance changes. The gradient-based features are affected by the presence of other objects near to the object of interest in the window and require massive calculations effort, and their computations can be time-consuming. The texture-based features have relatively less sensitive to changes in illumination, but they are extremely sensitive to irregular or anarchic backgrounds. It was shown that SFPF tracking techniques were able to track the moving object mostly correctly, accurately, and predict the trajectory if there are no disturbances. Hence, it is clear that to overcome this sensibility, ambiguity, and the defects associated with the features in different circumstances and environments, merging of the results obtained by every single descriptor from the three features should be made in an adaptive manner into one integral tracking algorithm. And this is a reasonable step as a middle solution to compensate for the features' partial sensibility.

The ultimate state estimate is calculated using the weighted average of individual estimates obtained by each feature. Testing and verification of the tracking accuracy of a single-feature and three fused features based on particle filters were achieved using a collection of synthetic and real traffic sequences scenarios containing typical perturbations (shadows, variable background, partial and full occlusions, maneuvering, etc.).

The fundamental novel idea behind that upholds the algorithm proposed in this thesis is built on the following assumptions:

1. Because of the feature's varying sensitivity to typical disturbances, the proper fusing of data obtained through the use of more than two fused features will be more robust than if only a single feature is used.
2. In order to conserve time in the computation, we have supposed that a relatively small number of particles have to be employed.
3. In contrast to more complex systems based on the use of Artificial Intelligence, this tracking algorithm does not require any prior learning. The object tracking success and adaptability in the tracking process are based on "online" measurements and the different ways in which they are combined.

5.1 Impacts of varying some algorithm's parameters

In this section we will study and discuss the effect of changing some important parameters on the proposed algorithm in the vehicle tracking process, such are the effects of:

- Number of particles (N),
- Spread radius of particles (DR), and
- Thresholds of average similarities (AS).

The sensitivity of the algorithm to the parameters will be studied and illustrated in two examples, one for our created artificial video sequences and one for real video sequences.

5.1.1 Impact on artificial video sequences

1) Effect of the increasing number of particles

The effect of altering the number of particles (N) on the tracking behavior and procedure could be seen by using two different values for the number of particles (N) in two different simulations run for the same scenario (Fig.5.1). In the first case, the number of particles is chosen to be equal to 100, and in the second case, the number of particles is chosen to be equal to 200. The obtained average similarity and RMS errors results for both cases can be seen in Fig.5.2 and Fig.5.3 respectively.

Table 5.I lists the obtained RMS errors for both cases using the SFPP and MFPP algorithms.

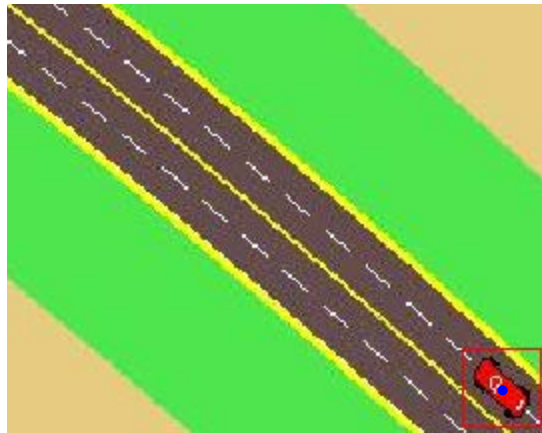


Fig.5.1 Artificial scenario for checking the impact of tracking algorithm parameters

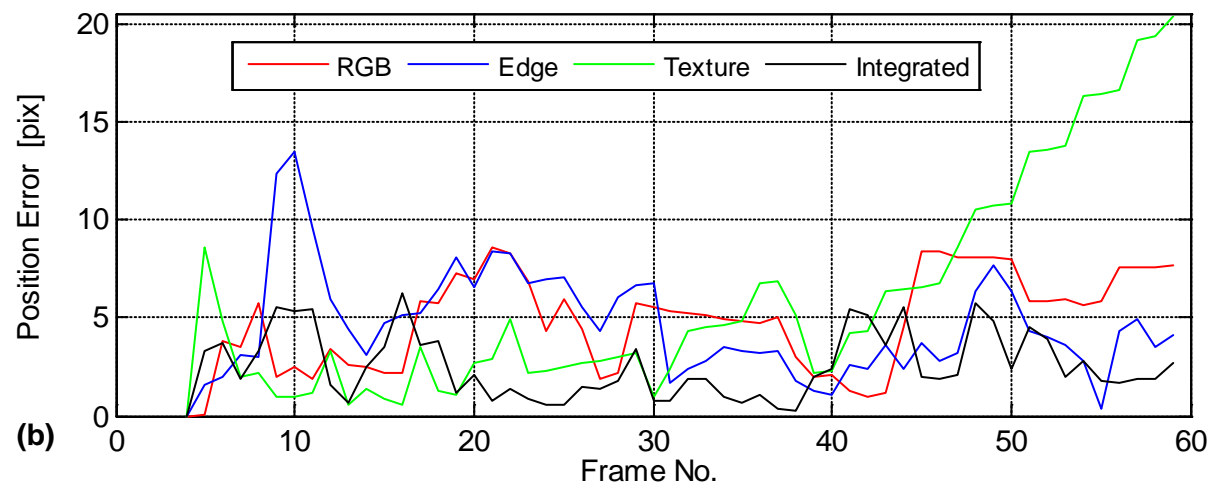
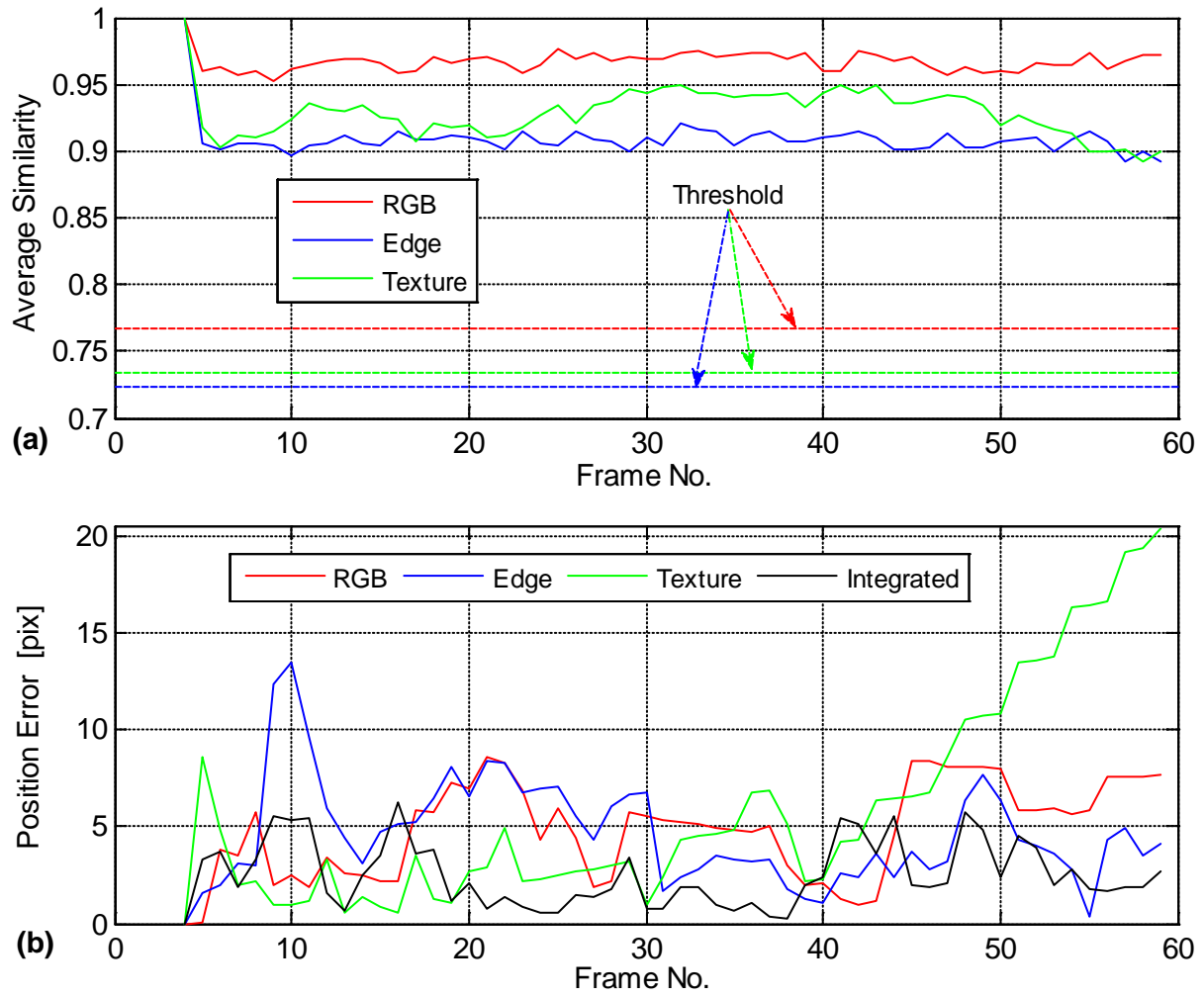
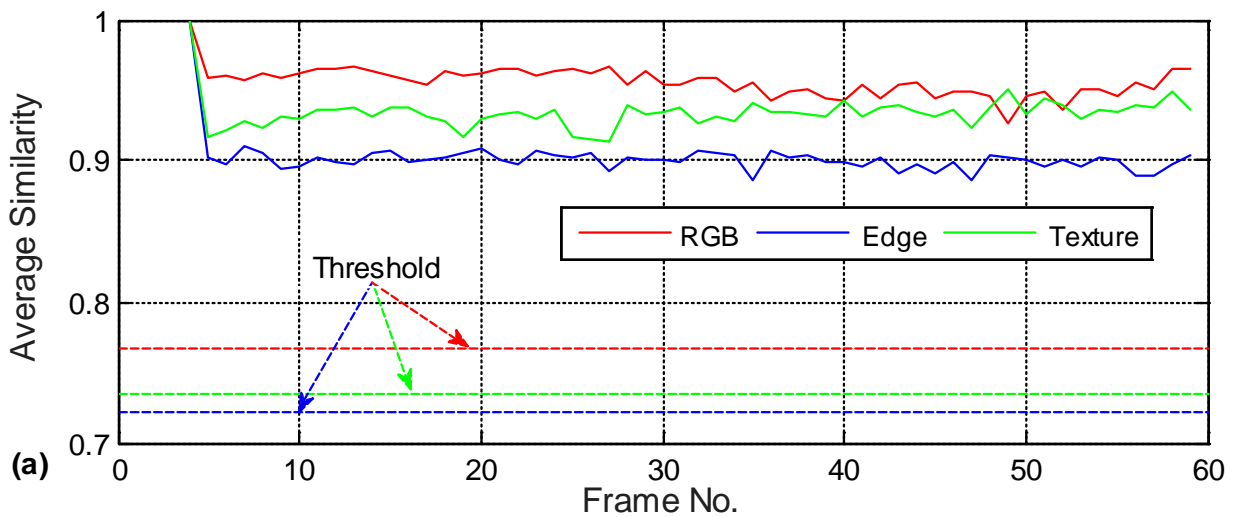


Fig.5.2 Case 1.: $N=100$ particles a) Average similarities, b) Tracking errors



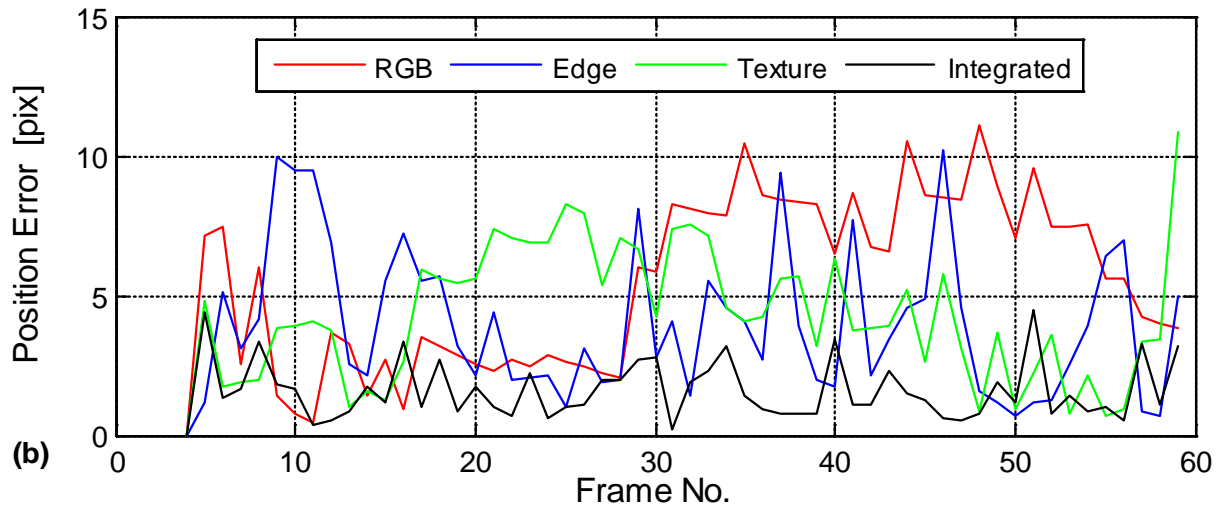


Fig.5. 3 Case 2: $N=200$ particles a) Average similarities, b) Tracking errors

Table 5.I RMS TRACKING ERRORS (PIXELS)

	RMS Error [pix]	
	N = 100 particles	N = 200 particles
RGB	5.41	6.19
Edge	5.35	4.76
Texture	8.04	4.88
Integrated	3.01	1.89

Regarding increasing the number of spread particles, it's clear from Table 5.I that the integrated RMS error is decreased as the number of spread particles is increased.

2) Effect of increasing the diffusion radius

The spread radius (DR) impact on the tracking behavior could be checked by selecting two values of DR for the same scenario in two different simulation runs. In the first case, DR was chosen to be equal to 10 pixels, and in the second case, DR was chosen to be equal to 15 pixels. The obtained average similarity and RMS error results for both cases can be seen in Fig.5.4 and Fig.5.5 respectively.

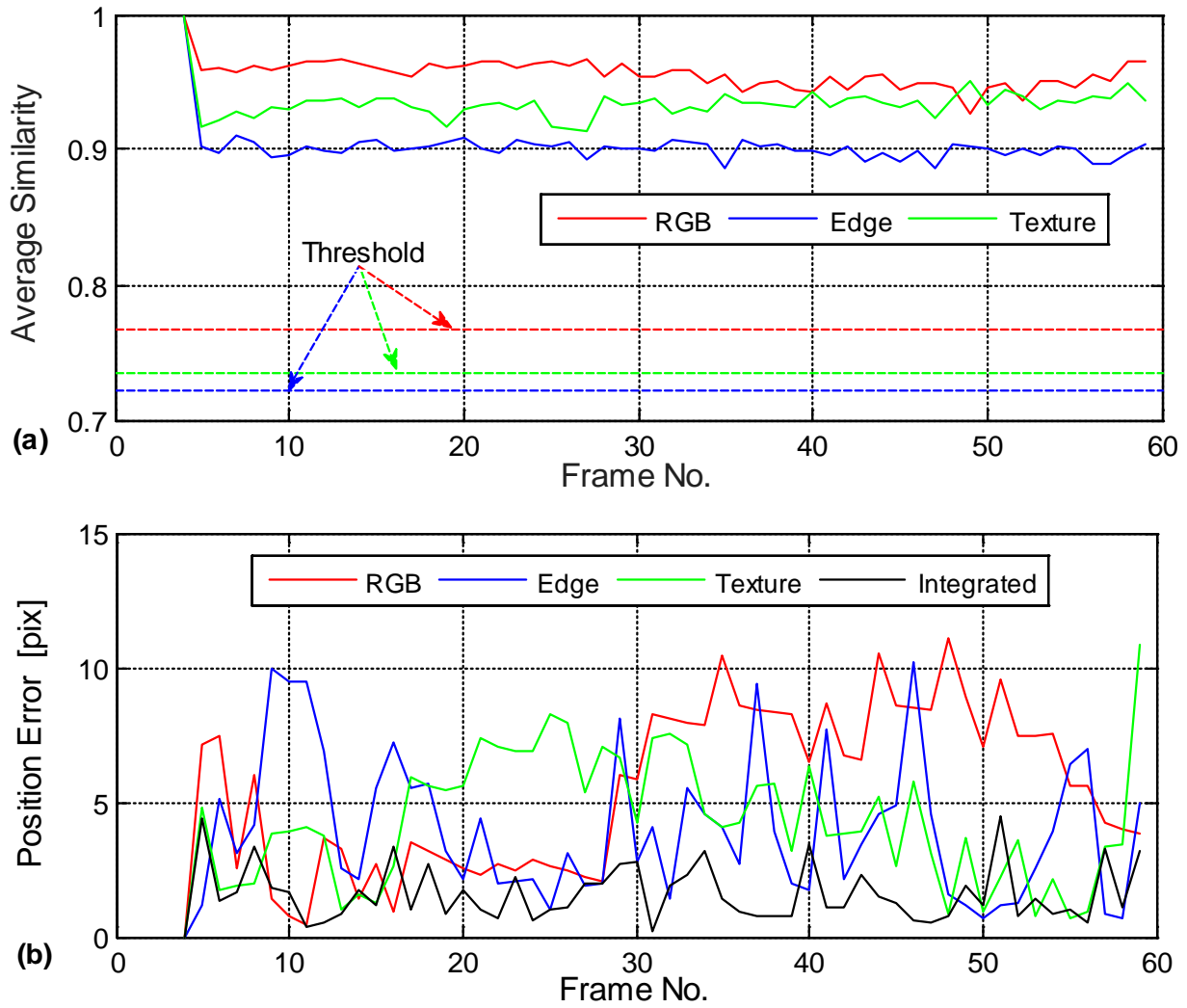
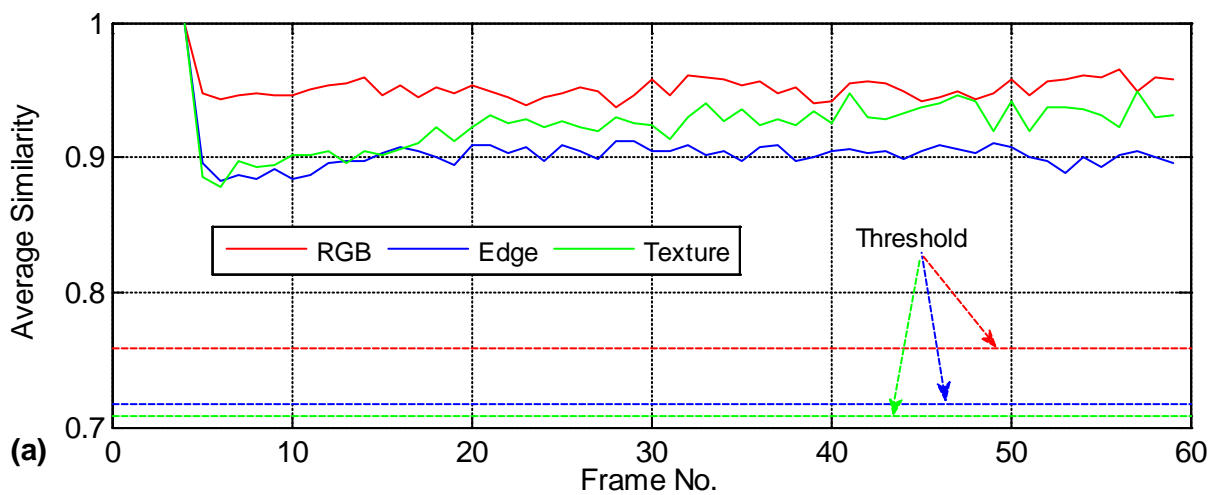


Fig.5.4 Case 1.: $DR=10$ pixels a) Average similarities, b) Tracking errors



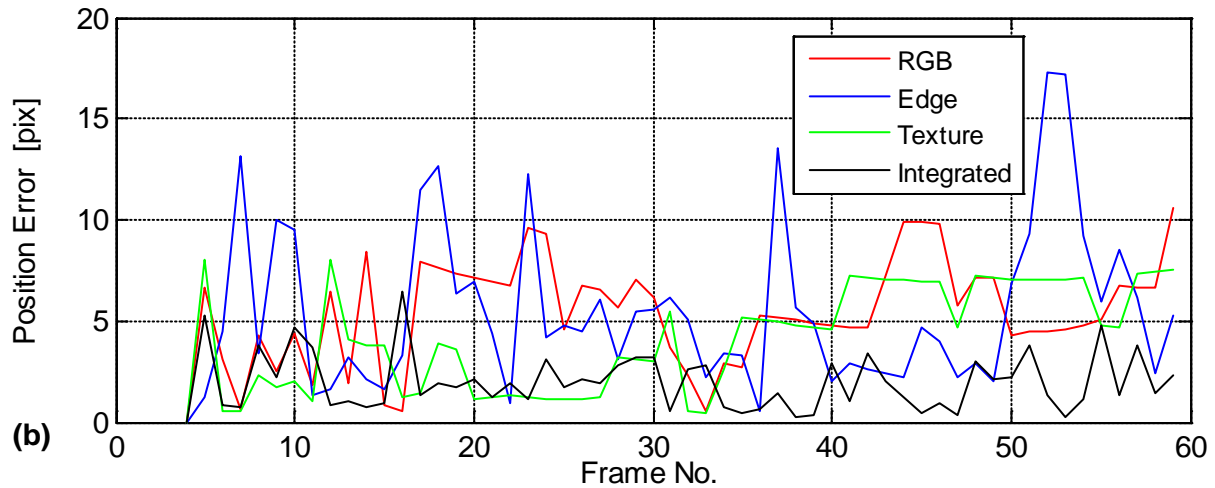


Fig.5. 5 Case 2: $DR=15$ pixels a) Average similarities, b) Tracking errors

Table 5.II illustrates the obtained RMS errors for the SFPF and MFPP algorithms for both cases.

Table 5.II RMS TRACKING ERRORS (PIXELS)

	RMS Error [pix]	
	RD = 10 pixels	RD = 15 pixels
RGB	6.19	5.98
Edge	4.76	6.76
Texture	4.88	4.86
Integrated	1.89	2.41

The effect of increasing/decreasing spread radiuses of particles in this particular artificial case as illustrated in Table 5.II is that, as the spread radiuses are increased, the integrated RMS error is increased.

3) Effect of thresholds of average similarities

To see the impact of thresholds of average similarities, there were considered two cases for the same scenario at different average similarity thresholds for each case.

In the first case, the average similarity threshold was selected to be equal to 60% of the initial average similarity in the first frame of the scenario for each descriptor, while in the second case, it was 80% of the first average similarity in the first frame. The obtained average similarity and RMS error results for both cases can be seen in Fig.5.6 and Fig.5.7 respectively.

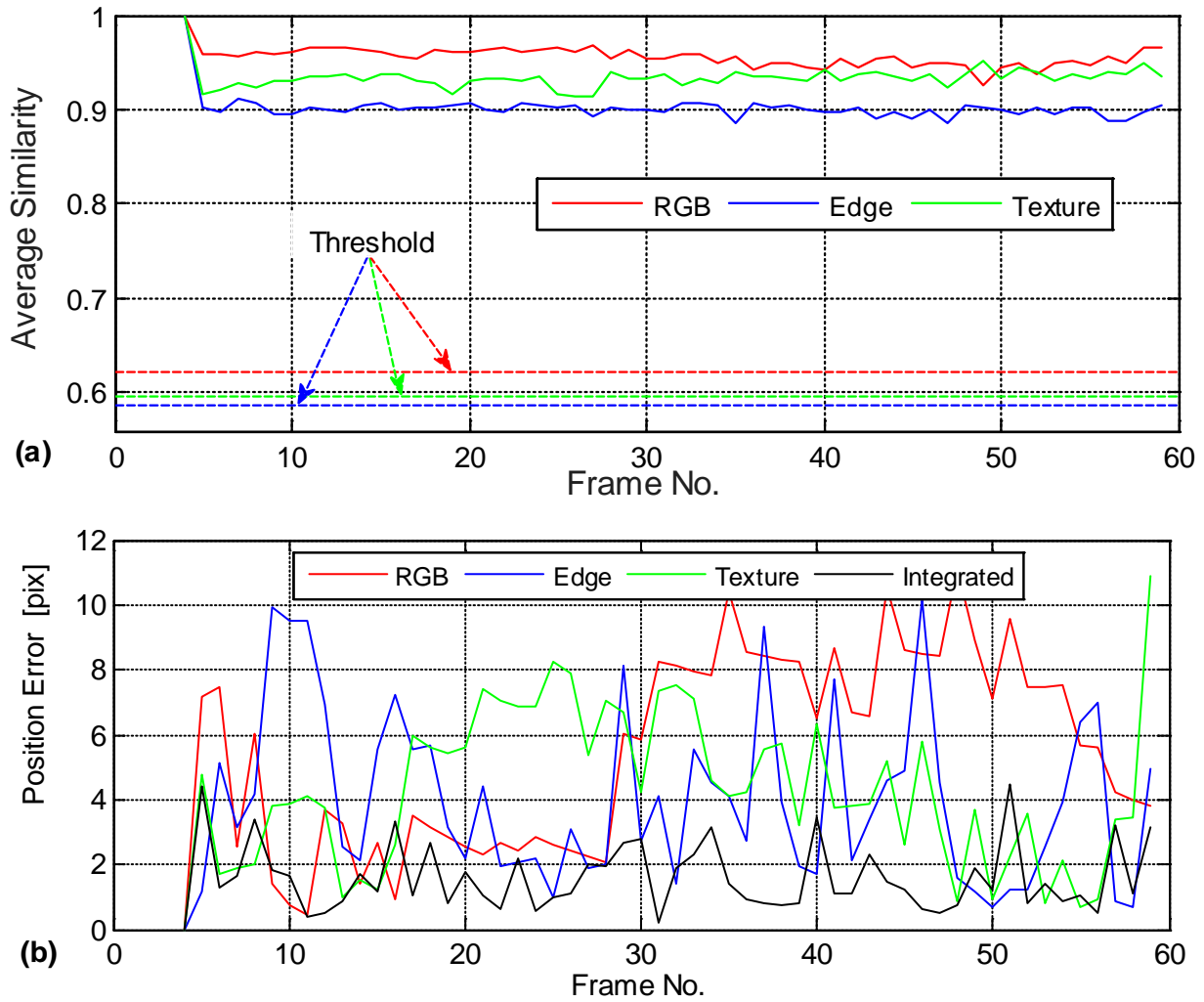
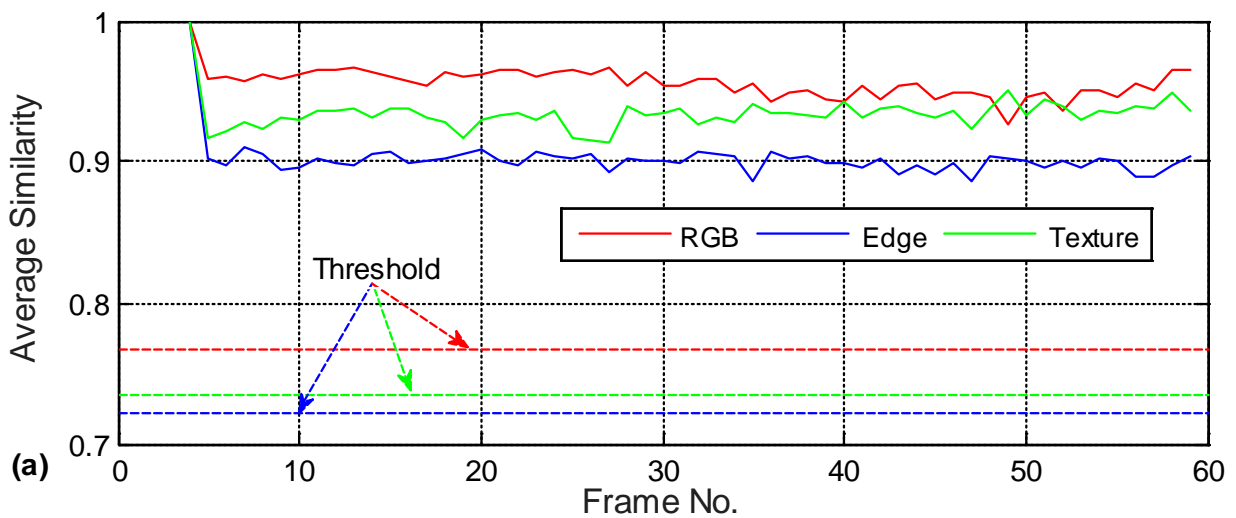


Fig.5.6 Case 1: Threshold=60% a) Average similarities, b) Tracking errors



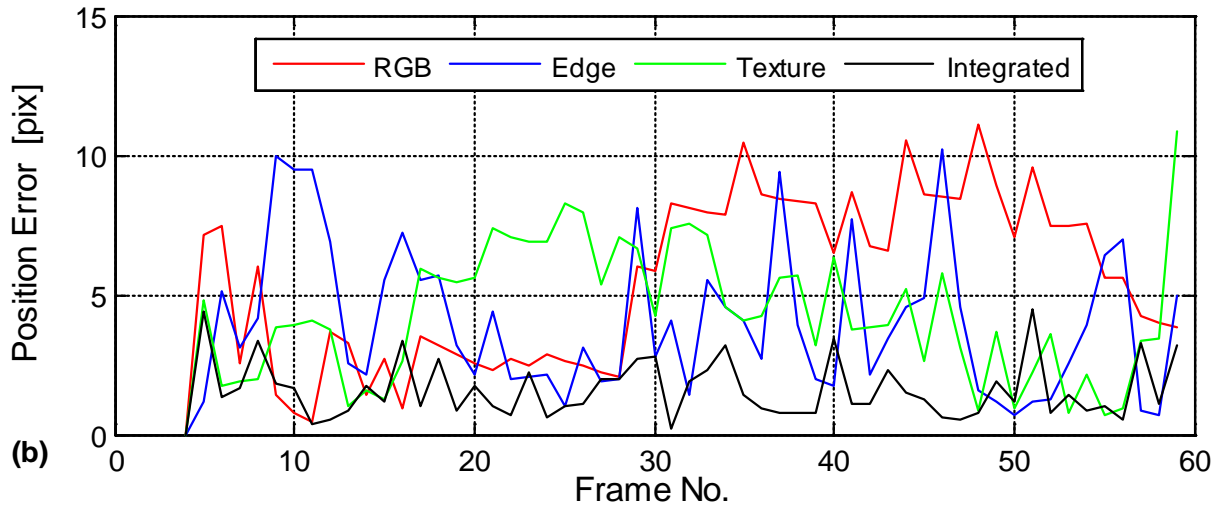


Fig.5.7 Case 2: Threshold=80% a) Average similarities, b) Tracking errors

Table 5.III represents the obtained RMS errors from the SFPP and MFPP algorithms.

Table 5.III RMS TRACKING ERRORS (PIXELS)

	RMS Error [pix]	
	Threshold = 60%	Threshold = 80%
RGB	6.19	6.19
Edge	4.76	4.76
Texture	4.88	4.88
Integrated	1.89	1.89

Looking at Table 5.III, one can see that the RMS errors are not affected in both cases by the two given thresholds (60% or 80% of the first value of average similarity).

5.1.2 Impact on real video sequences

We have selected the scenario shown in Fig.5.8 for evaluating the effects of parameters on the proposed tracking algorithm.



Fig.5.8 Selected real scenario for checking the parameters on the proposed tracking algorithm

1) Effect of the increasing number of particles

The effect of the number of particles on the tracking behavior is shown in the obtained results of average similarity and RMS errors for both cases as shown Fig.5.9 and Fig.5.10 respectively, and Table 5.IV below lists the obtained RMS errors for the SFPF and MFPP algorithms for each case.

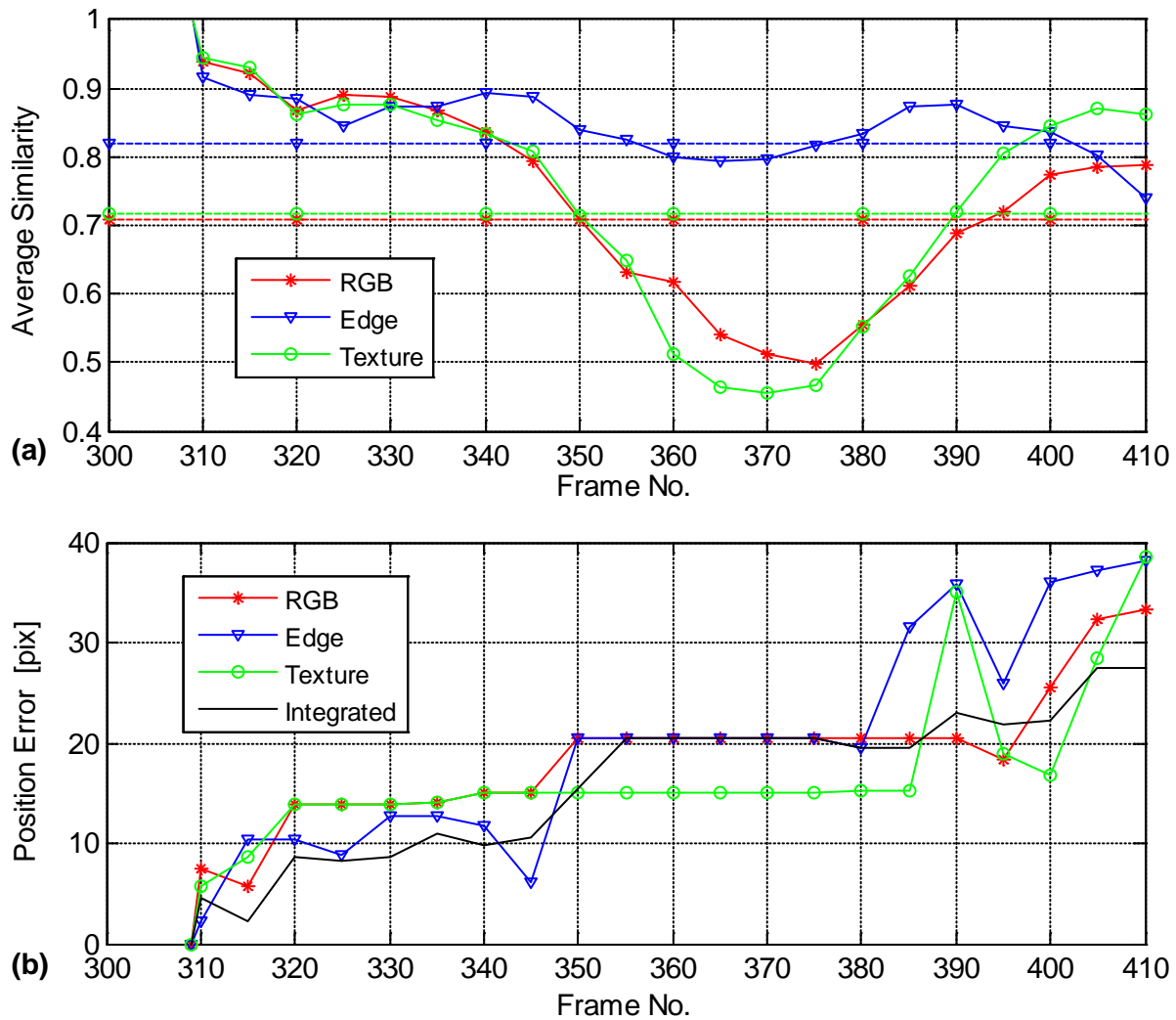


Fig.5.9 Case 1.: $N=100$ particles a) Average similarities, b) Tracking errors

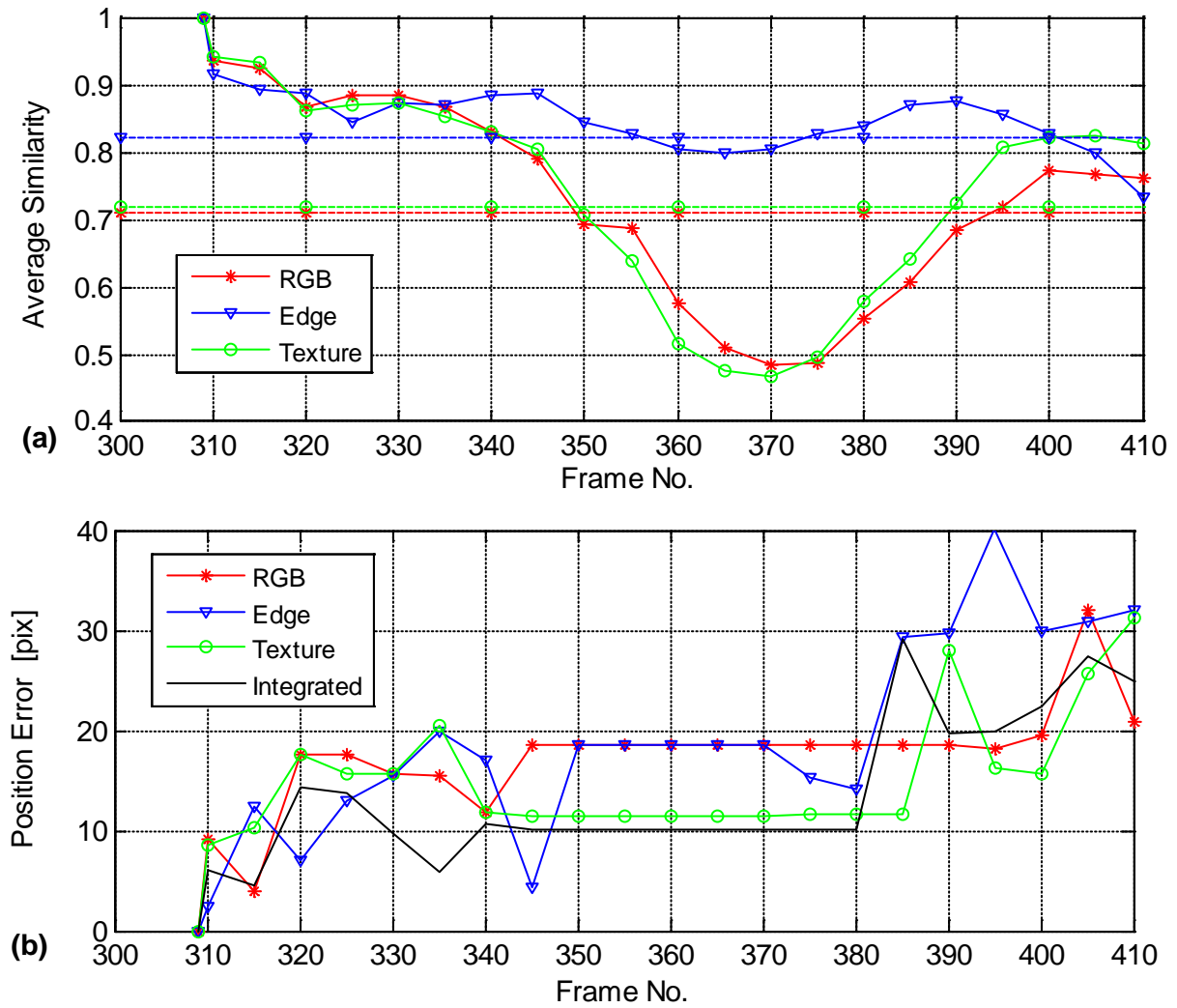


Fig.5.10 Case 2: $N=200$ particles a) Average similarities, b) Tracking errors

Table 5.IV RMS TRACKING ERRORS (PIXELS)

	RMS Error [pix]	
	N = 100 particles	N = 200 particles
RGB	19.41	17.81
Edge	22.16	21.02
Texture	18.26	16.04
Integrated	17.45	15.11

Table 5.IV shows that the integrated RMS error is decreased as the number of spread particles are increased.

2) The Effect of increasing the diffusion radius

The spread radius (DR) chosen for both cases is 15 pixels and 20 pixels respectively. The impact of varying the spread radius on the tracking behavior in this scenario in both cases could be seen in the obtained results of the average similarity and the RMS error results as seen in Fig.5.11 and Fig.5.12 respectively.

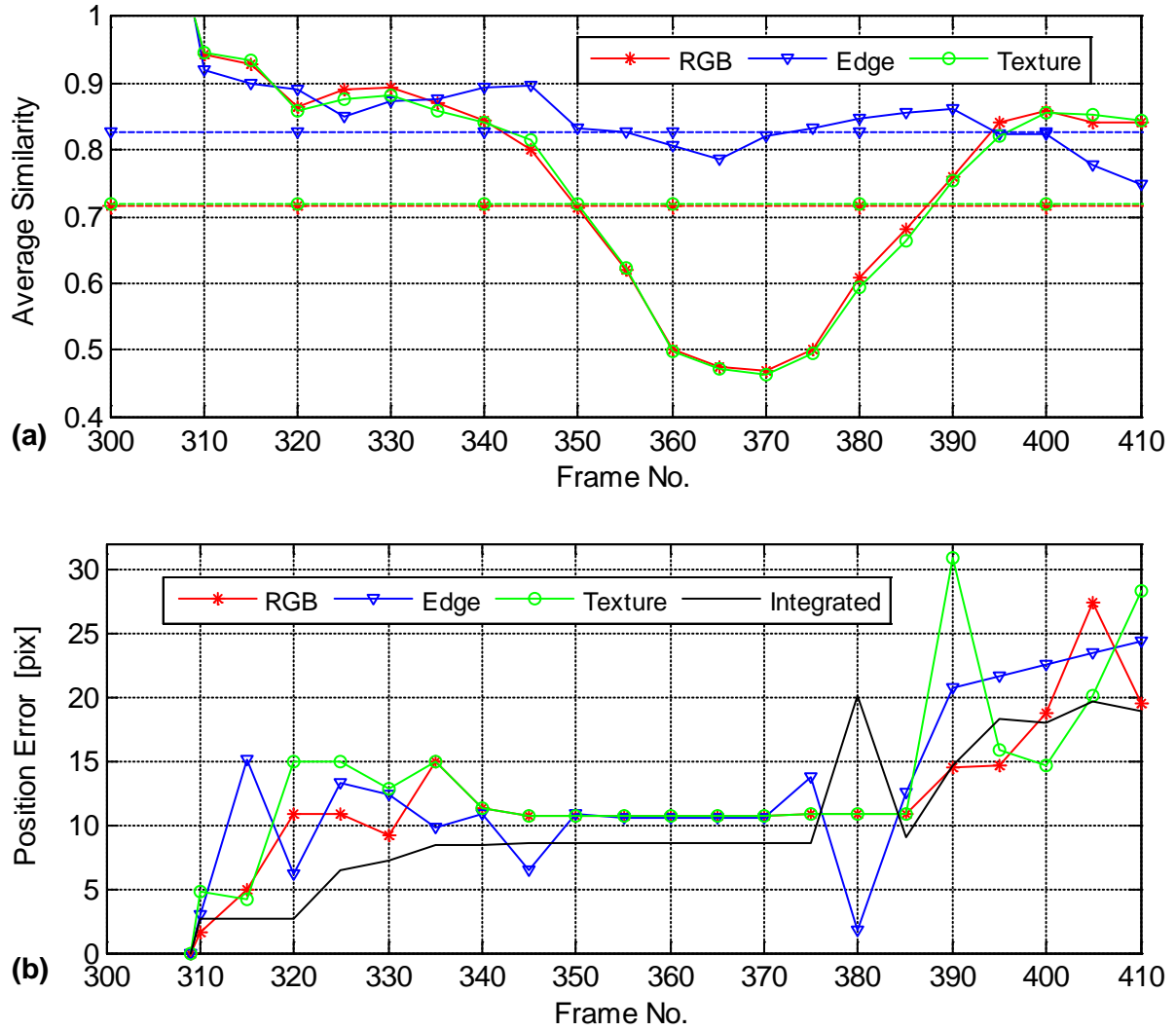


Fig.5.11 Case 1: DR=15 pixels a) Average similarities, b) Tracking errors

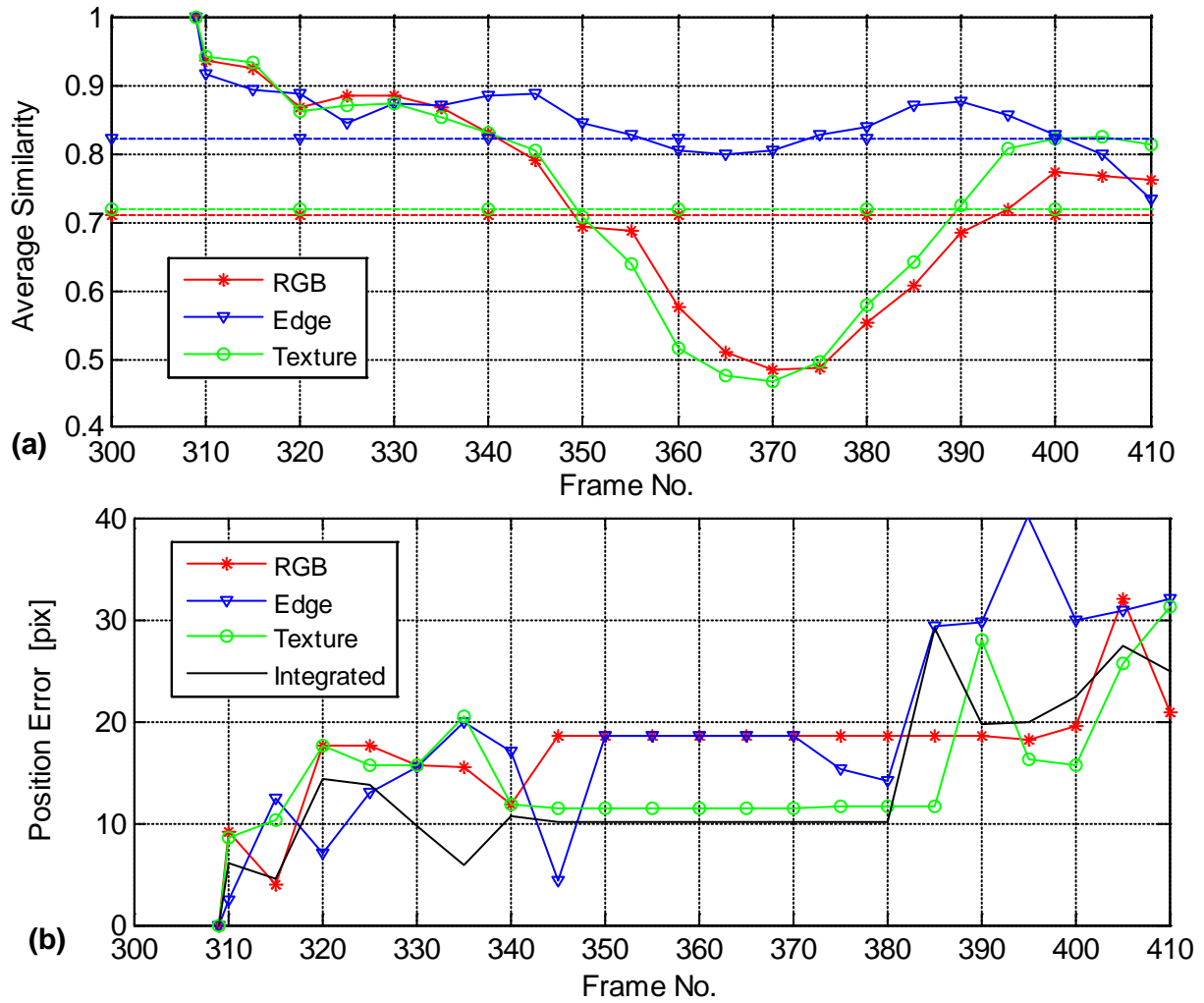


Fig.5.12 Case 2: DR=20 pixels a) Average similarities, b) Tracking errors

Table 5.V below lists the obtained RMS errors for the SFPF and MFPP algorithms for both cases.

Table 5.V RMS TRACKING ERRORS (PIXELS)

	RMS Error [pix]	
	RD = 15 pix	RD = 20 pix
RGB	12.87	17.81
Edge	13.98	21.02
Texture	14.55	16.04
Integrated	11.41	15.11

In this particular case of real video, Table V illustrated that the RMS errors increased as the spread radiuses increased.

3) Effect of average similarity thresholds

The effect of similarity thresholds on the considered scenario was tested by applying two different similarity thresholds: 60% and 85% of the initial average similarity in the first frame of the scenario. The impact of changing the similarity threshold for both cases could be seen in Fig.5.13 and Fig.5.14 respectively.

Table 5.VI summarizes the obtained RMS errors in the SFPF and MFPP algorithms for each case.

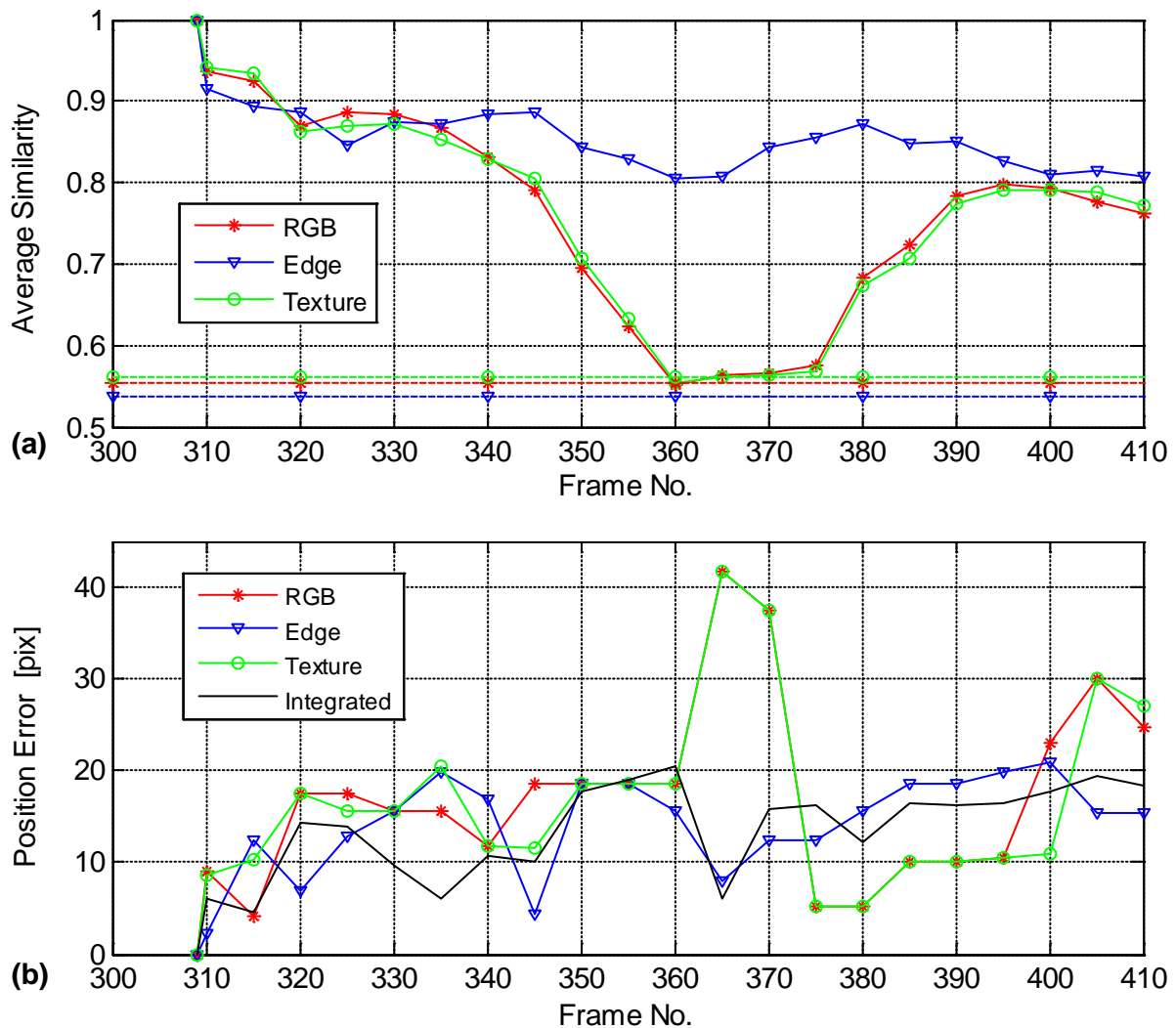


Fig.5.13 Case 1: Threshold=60% a) Average similarities, b) Tracking errors

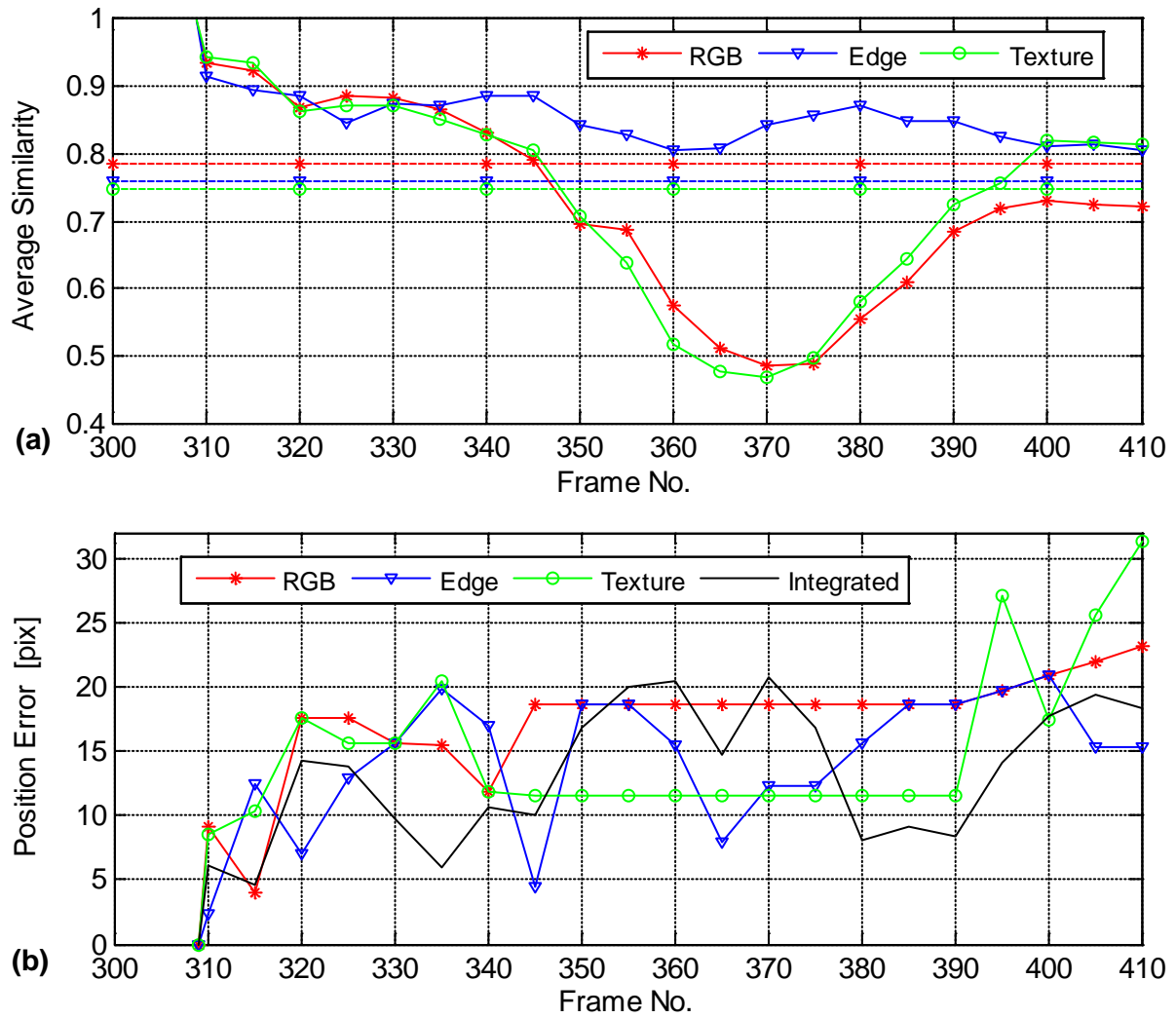


Fig.5.14 Case 2: Threshold= 85% a) Average similarities, b) Tracking errors

Table 5.VI RMS TRACKING ERRORS (PIXELS)

	RMS Error [pix]	
	Threshold = 60%	Threshold = 85%
RGB	19.39	17.39
Edge	14.85	14.85
Texture	19.01	15.86
Integrated	14.18	13.93

Table VI shows that the two cases of thresholds (60% or 85% of the first value of average similarity) give almost the same RMS error in this real video.

5.2 Experimental Comparison of MFPP and Different SFPP Algorithms

Several scenarios for testing and analyzing vehicle tracking have been used in this thesis. Some of them are artificial and others are real. We have picked some of them for illustration and discussion and divided them into two groups. The first group is the created synthetic video sequences (sec. 5.2.1), and the second group is the real video sequences (sec. 5.2.2). And each group is categorized into cases, and each case is considered as a specific issue. In all cases, the illustration of the entire scene is always shown first, followed by a collection of frames throughout the tracking process. Average similarities of the proposed three features for the tracked vehicle's window are displayed as curves with their predefined threshold values. The threshold is the reference used to point out that if the average similarity value of a feature is less than the threshold value, then a feature becomes worthless for tracking and is eliminated from the estimate process. The other kinds of curves are used to indicate how the tracking errors are variable (difference between estimated positions using the three features and the actual vehicle's position as determined by data in pixels). These data are used to compute the Root Mean Square (RMS) tracking error that is indicated in the equation (5.1).

$$RMS = \sqrt{\frac{1}{i_{\max}} \sum_{i=1}^{i_{\max}} [(x_{es} - x_{tr})^2 + (y_{es} - y_{tr})^2]} \quad (5.1)$$

where:

- i_{\max} , refers to a set of validation points;
- x_{es}, y_{es} , are the estimated coordinates of the object in pixel, and
- x_{tr}, y_{tr} , are the real object coordinates in pixels.

The root mean square errors for the artificial and the real video sequences are shown in Tables 5.VII and 5.VIII in the next section. RMS is used to compare the tracking **accuracy** of the three SFPPs and the proposed MFPP.

5.2.1 Synthetic video sequences

The assumed parameters for synthetic video sequence simulations are as follows:

- The time interval between frames (IBF) – 50 ms,
- Velocity of the tracked vehicle – 90 km/h (25 m/s, 1.25 m/IBF),
- The number of particles used in PF – 150,
- The number of particles used in PF was 150 in Case-I, Case-II, and 200 in Case-III.
- The spreading radius of the particles was 10 pixels in Cases-I and II, and 15 pixels in Case-III.
- Tracking window – 32×32 pixels,
- 1 meter \approx 7 pixels.

1) Case-I: Tracking the target car by fusion of multi-features without disturbances

Fig.5.15 shows a simple example of synthetic sequences representing the tracking of a car that is not exposed to any type of disturbances during the scenario.

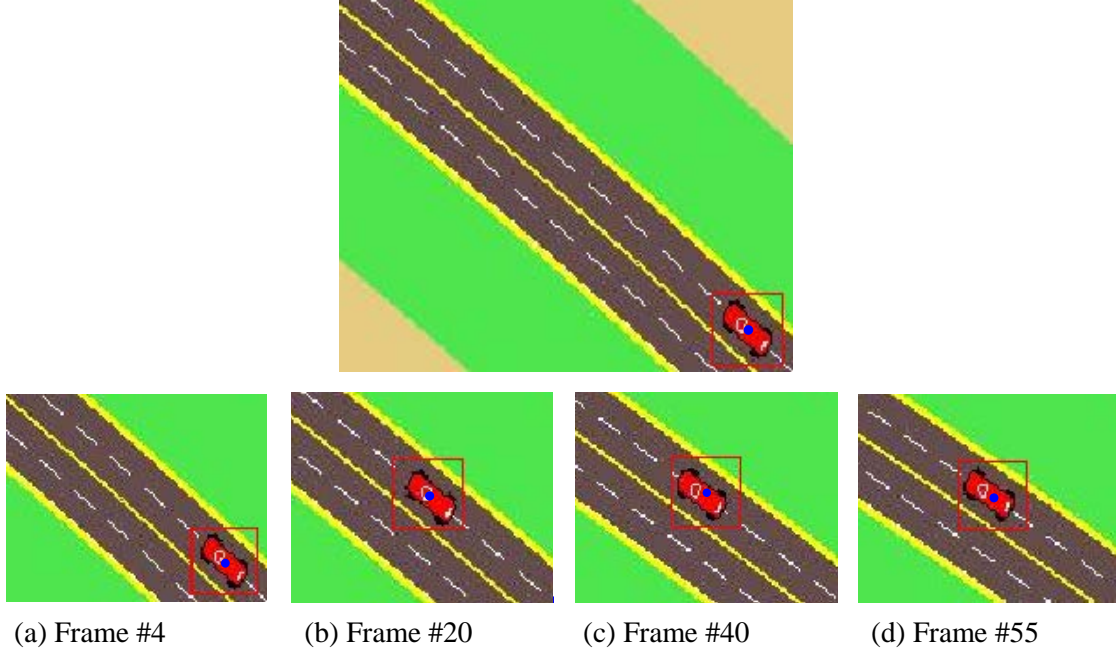
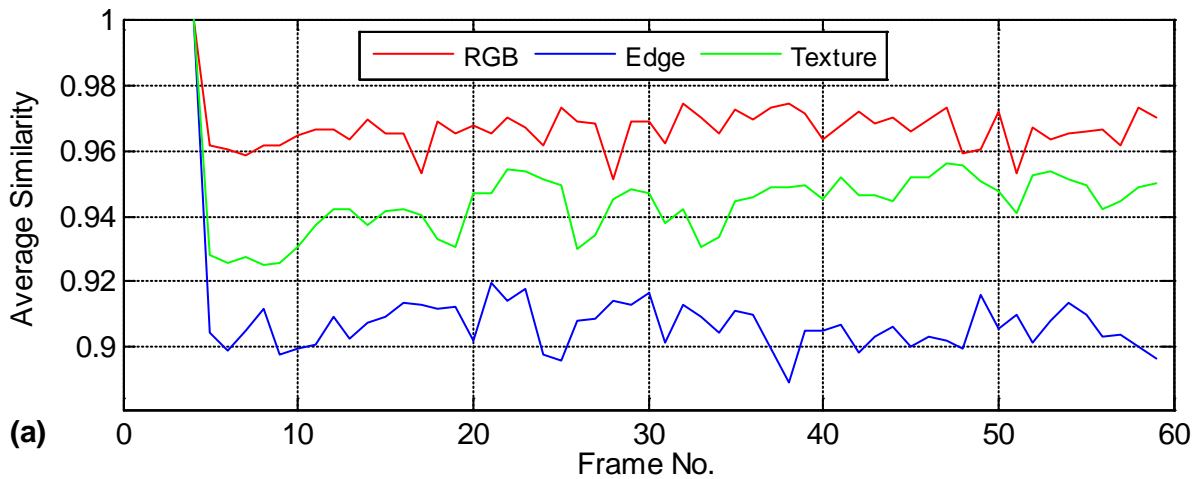


Fig.5.15 Tracked car on the road

Fig.5.16 shows the average similarity measures (Fig.5.16.a) and the position errors between the estimated car positions and true ground position (Fig.5.16.b) using the three proposed descriptors separately and integrated as well.



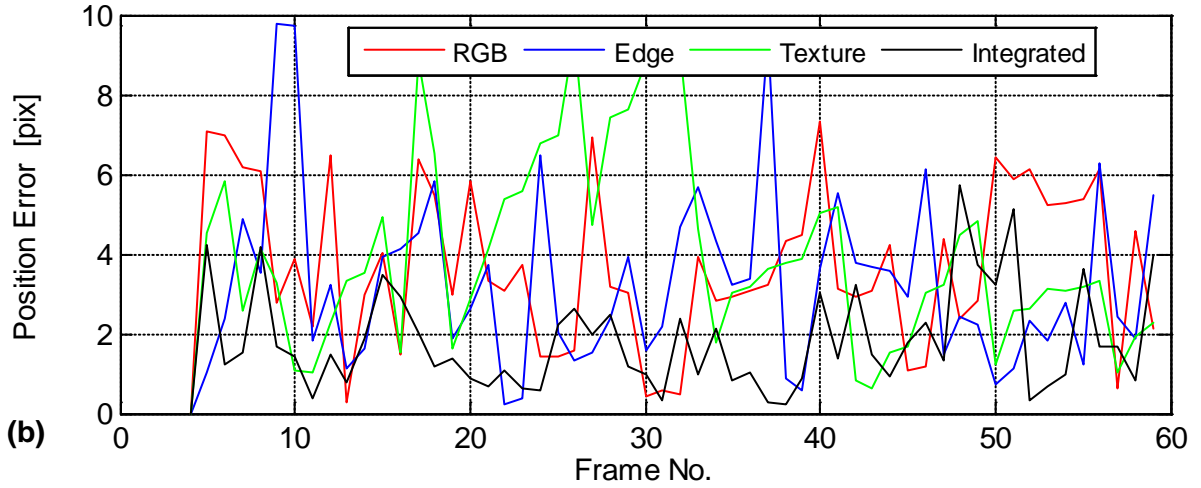


Fig.5.16 Case-I a) Average similarities, b) Tracking errors

From Fig.5.16.a one can see that the average similarity gives preference to color feature. The RMS tracking errors for the three individual descriptors are 4.18, 3.93, and 4.61, while for the integrated algorithm, the error is 2.21 pixels.

2) Case-II: Tracking in the presence of the car's self-shadow and shadow produced by the road environment

The disturbances such as self-shadow, shadows created by trees and buildings or clouds are included in this example of synthetic sequences, and these disturbances appear abruptly at different times.

Fig.5.17.b presents the car with its self-shadow (frame #15), while Fig.5.17.c presents the presence of a shadow produced by nearby buildings (frame #45).

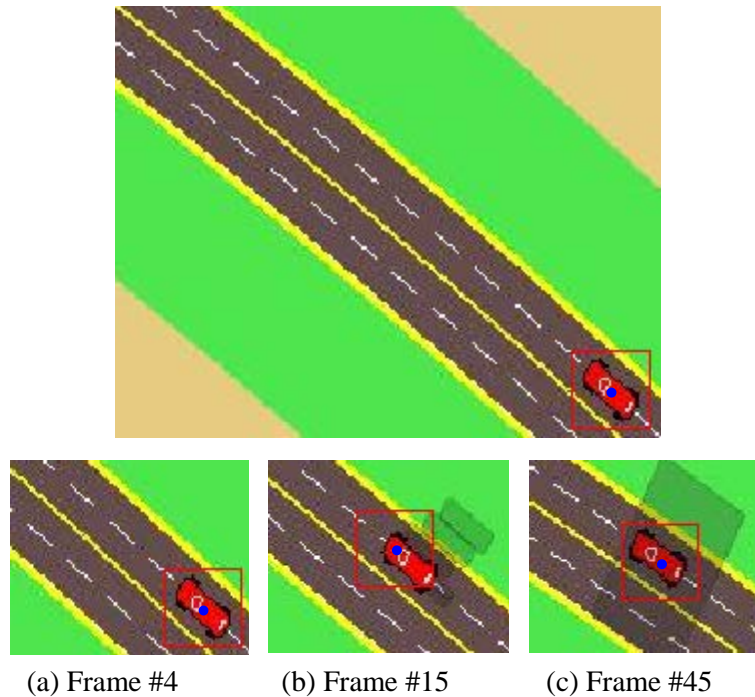


Fig.5.17 Case-II a) Reference frame (#4), b) Self-shadow effect (#15), c) The effect of shadow produced by environment objects (#45)

The average similarity and the deviation between the actual and estimated (updated) vehicle's position have been presented in Fig.5.18 below. The error curves (Fig.5.18.b) show that all features have been influenced by the presence of the vehicle's self-shadow (frames 13 to 28) and the building's shadow (frames 42 to 48).

We can see from Fig.5.18.a that the texture's average similarity is less than the specified similarity threshold in the period 13 to 42. Therefore it is excluded from the tracking process during this time period (frames), so the tracking relies on the color and edge features only. In addition, we can obviously see that the color and the texture descriptors' average similarity are sharply decreased within the period 42 to 48 (the period of building shadow or road environment shadow). And both descriptors have been excluded because their associated similarity values are less than the predefined similarity thresholds as shown in Fig.5.18.a. As a result, the tracking process is solely focused on the edge feature.

The RMS tracking errors for the three separate features and integrated algorithms are 6.94, 8.19, 5.61, and 4.36 pixels respectively.

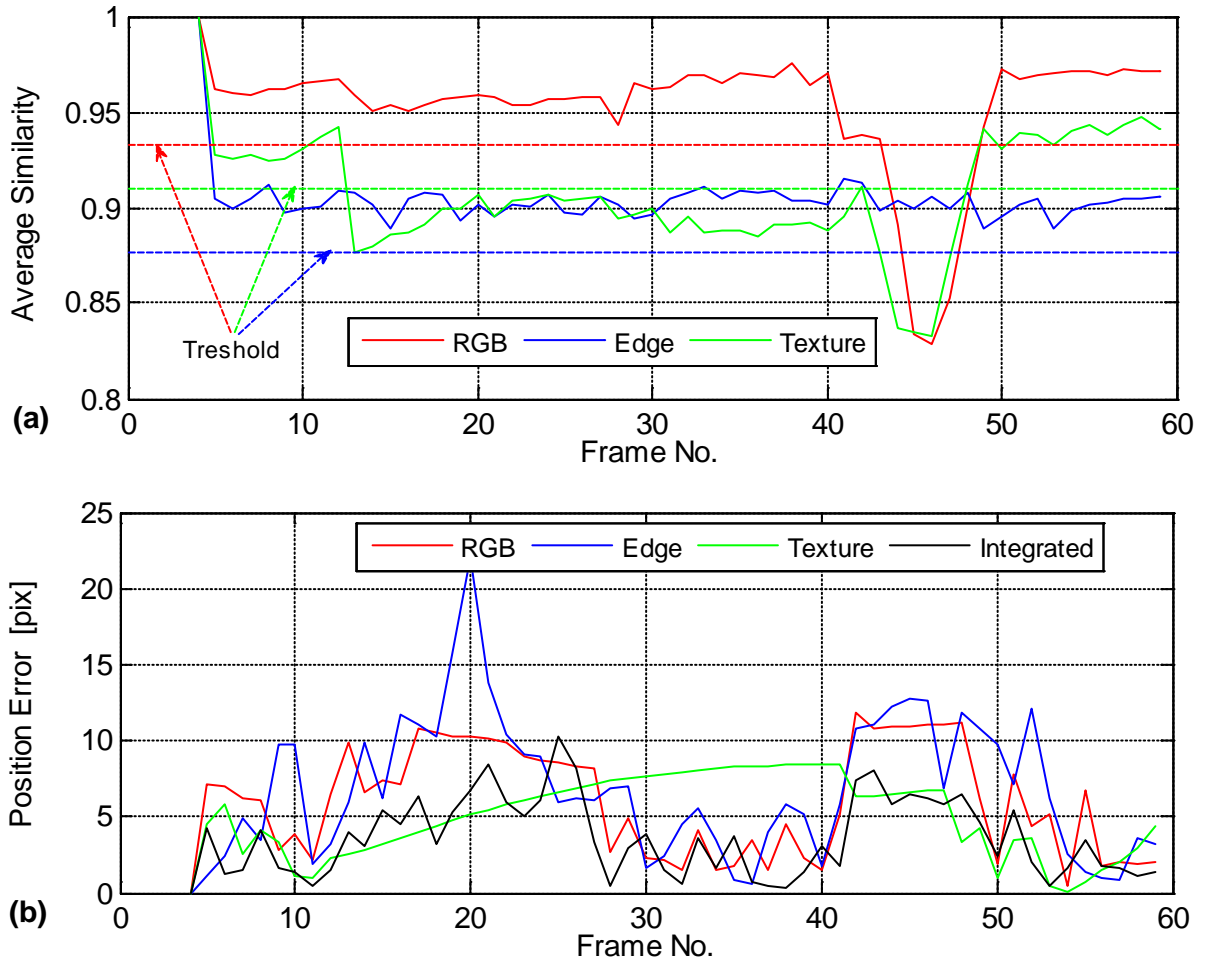


Fig.5.18 Case-II a) Average similarities, b) Tracking errors

3) Case-III: Tracking of a target vehicle during a maneuver

Fig.5.19 illustrates the artificial tracking scenario, that includes the phases of an ego car (the red one) bypassing the other car (the white one).

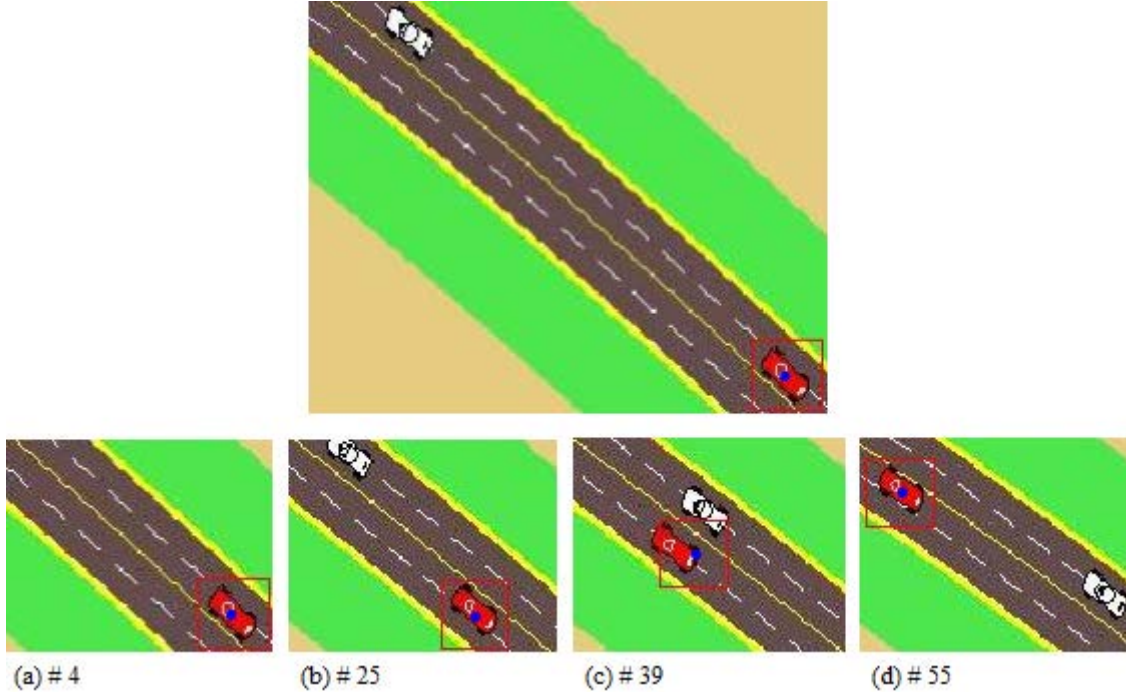
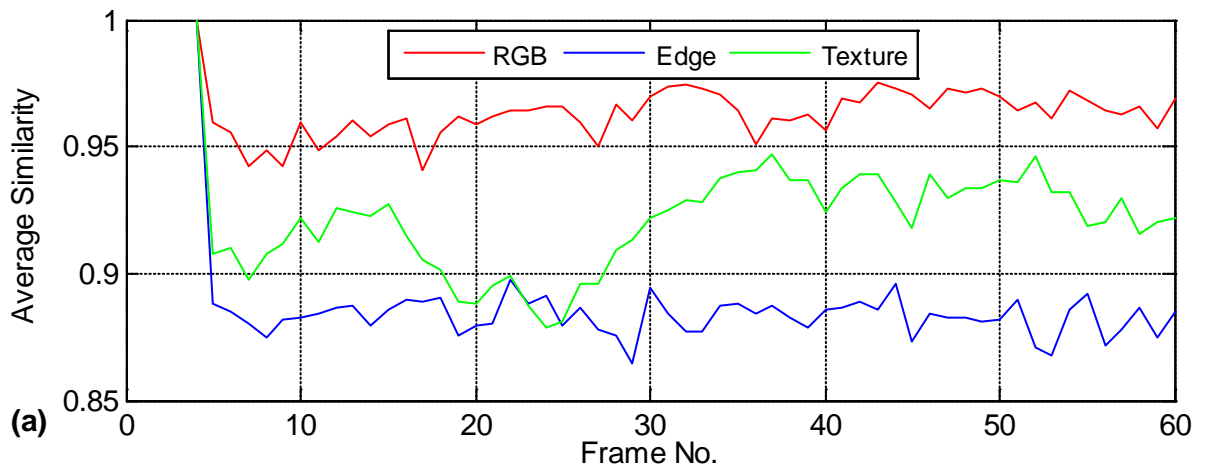


Fig.5.19 Case-III: Over passing status frames, a) Reference frame (#4), b) before over passing (#25), c) two cars in parallel (#39), d) after over passing (#55)

The maneuvering phases in this scenario took place between frames #4 and #60. The average similarities and the tracking errors are presented in Fig.5.20 below.

It is obvious from Fig.5.20.a that the color feature was the least sensitive and it has the highest average similarity.

The obtained RMS tracking errors for three individual descriptors and integrated PF algorithms are 6.67, 11.26, 9.92, and 5.25 pixels, consecutively.



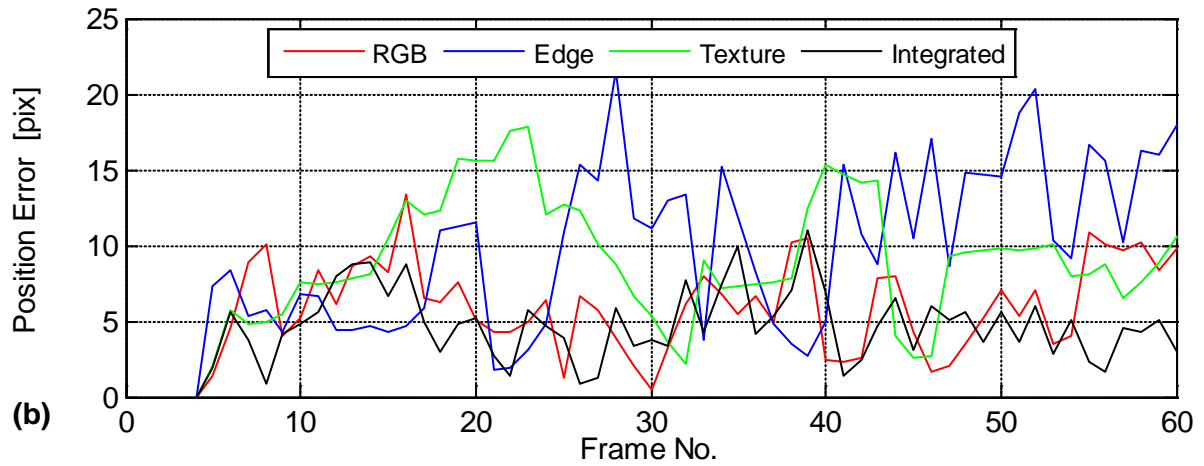


Fig.5.20 Case III a) Average similarities, b) Tracking errors

By comparing all results obtained for the simple artificial sequences (Cases I to III) with the isolated disturbances, one can conclude that the “Integrated filter” was the best in all cases. Therefore, there is no possibility to decide which separate PF is the best candidate for all cases. Accordingly, the following sets of real traffic sequences in the next section are analyzed, where all typical disturbances act together simultaneously.

Table 5.VII illustrates the RMS errors in pixels for all three previous examples of artificial sequence cases. Analyzing the results obtained for artificial sequences (Cases I-III), one can see the superiority of the integrated tracking algorithm compared to any of the individual features separately.

Comparing the results obtained from the SFPP, one can conclude that there is no particular SFPP that could be declared as the best in all cases. There are cases where each one of them might be the best candidate, and at the same time, this SFPP could be the worst in some other scenarios.

Table 5.VII RMS tracking errors in pixels

		RMS Error [pix]		
		Case I	Case II	Case III
SFPP	RGB	4.18	6.94	6.67
	Edge	3.93	8.19	11.26
	Texture	4.61	5.61	9.92
MFPP		2.21	2.21	5.25
MFPP Relative to the best SFPP [%]		56.23	77.72	78.71

Table 5.VII shows that the three-feature-based filter that uses the image's three features gives RMS tracking errors that are 21% – 43% less than the lower errors among the SFPP RMS tracking errors.

5.2.2 Real video sequences

Some of the selected parameters for the real video sequences have been changed in comparison to artificial video sequences. The spreading radius of particles was selected to be 20 pixels because the size of the vehicles in the selected real scenarios was bigger. The number of particles used was 200 particles. The weighting coefficient (α) is 0.5. The sizes of tracking windows for the chosen six real scenarios cases have been: 168×96 , 50×40 , 120×110 , 65×45 , 100×72 , and 60×64 pixels respectively.

1) Case-IV: Influence of Small Partial Occlusion

This simple example figures out the partial occlusion of the vehicle caused by the existing trees beside the road, which appear at different intervals, as is illustrated in Fig.5.21. The obtained average similarities and tracking errors for this scenario can be seen in Fig.5.22.



Fig.5.21 Illustration of Case IV, frame (#15) is a reference frame

As shown in Fig.5.22.a, average similarities for the “Edge” and “Texture” features are always above their threshold values, whereas the “RGB” feature is the most affected. It is even eliminated from the estimating process in the last phase of the sequence. From Fig.5.22.b, it is clear that the “Integrated” filter is not mostly superior in any isolated particular frame in comparison to particular SFPFs. However, we can see that the RMS tracking error in the case of MFPP during the entire interval is the least, as seen in the first column of Table 5.VIII.

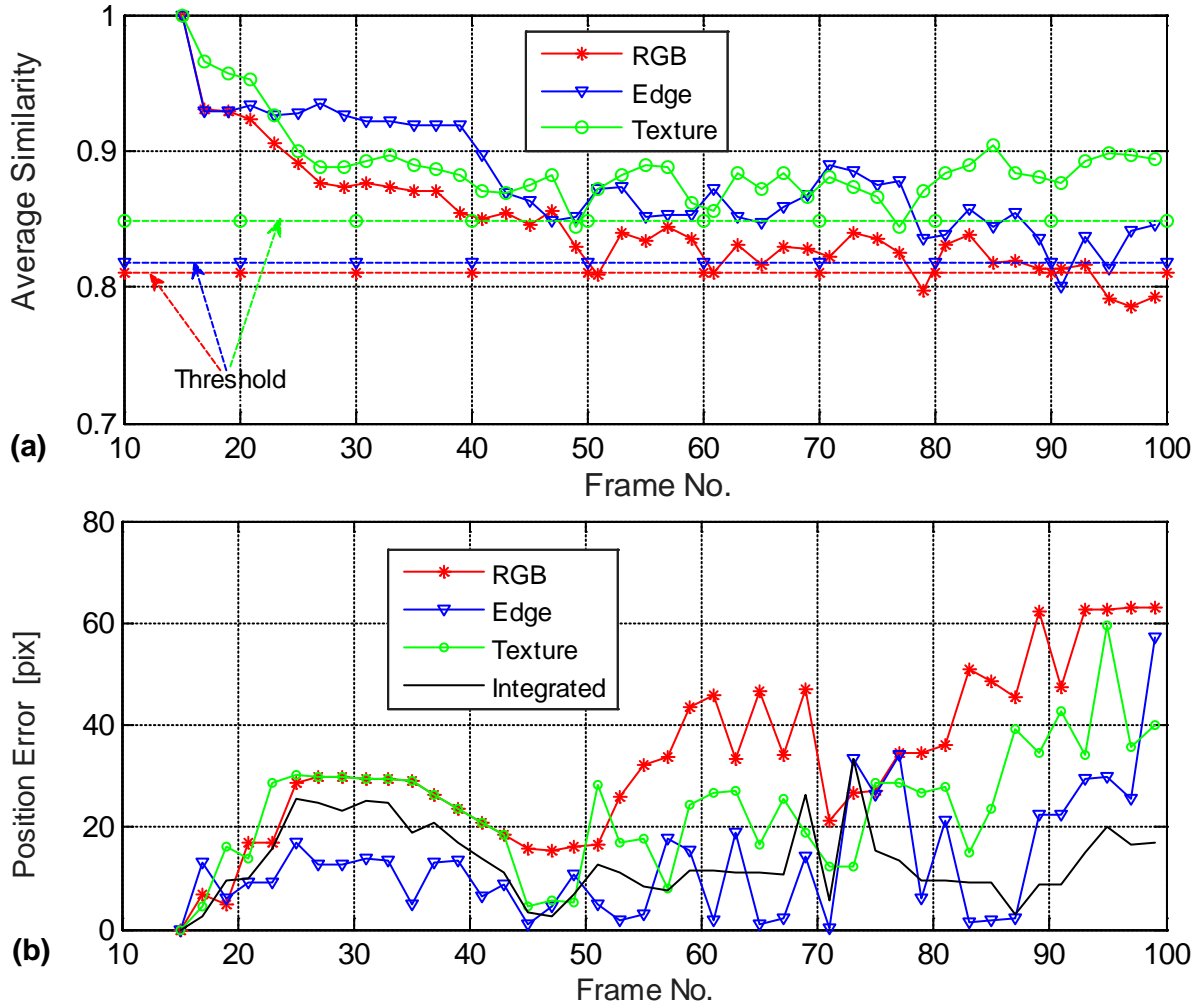


Fig.5.22 Case-IV: (a) Individual average similarities, (b) Individual SFPP errors Vs. MFPP error

2) Case-V: Influence of Longer Partial Occlusion

In this illustrative example, the arriving car is not wholly obscured (partial occlusion) when it is overtaking another vehicle at some period during the sequence of frames, as is seen in Fig.5.23.



Fig.5.23 Illustration of Case V

For the reason of low initial values of average similarities for the proposed three features, the suitable threshold values are comparatively low, and the threshold is specified as a percentage (e.g. 80%) of the initial values of average similarities. Practically, all the time maintaining all of the SFPPs as components in the estimate operation (see Fig.5.24). The most sensitive feature is the edge, taking into account that the contour orientations of the tracked vehicle also discriminate the vehicle that is being overrun. Table 5.VIII's second column demonstrates that the accuracy of all three SFPPs are extremely close, whereas MFPP is somewhat better.

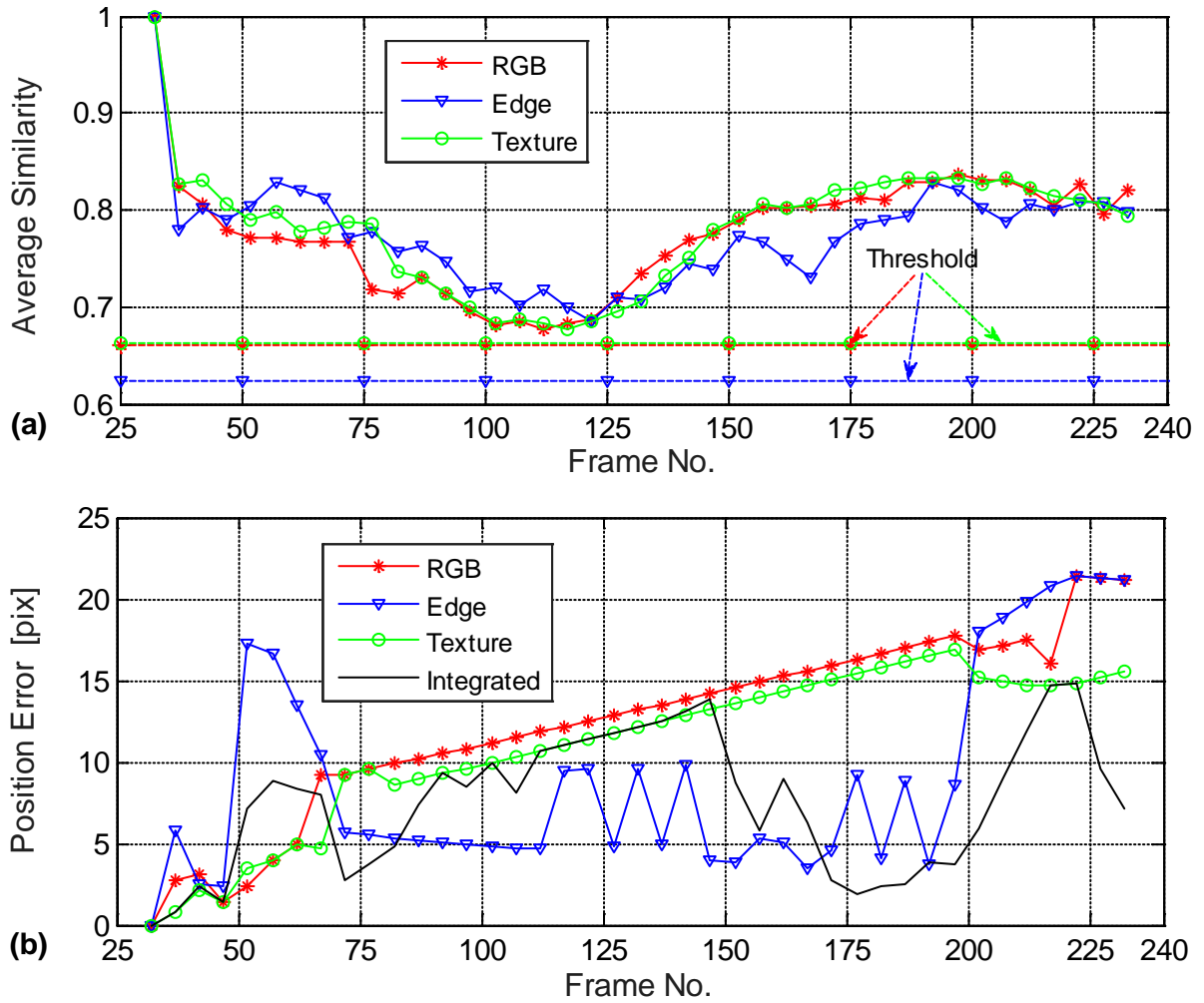


Fig.5.24 Case V: (a) Individual average similarities, (b) Individual SFPP errors Vs. MFPP error

3) Case-VI: Influence of Shadows

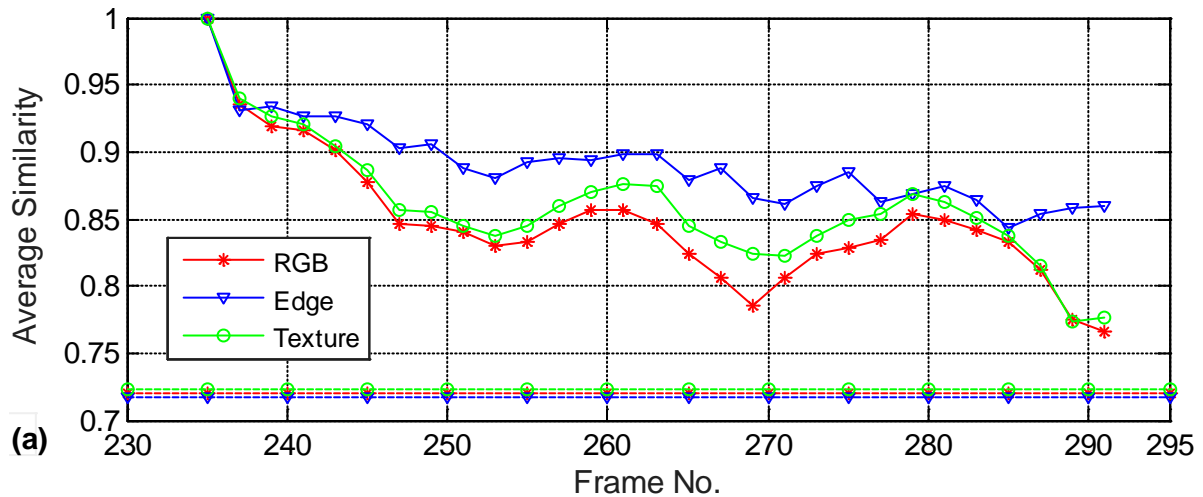
This example illustrates the circumstance in which the background is variable due to the presence of trees' shadows at different periods, as we can see in Fig.5.25.



Fig.5.25 Illustration of Case VI

Because some parts of the road are mostly darkened, and the edge feature is the feature that relies on the actual amount of shadow, and in addition to the changing (increase) of car size in the scenario, the “Edge” feature utilized by SFPF is mostly the sensitive one compared to the other features, as shown in Fig.5.26.b. The average similarities of all three features as shown in Fig.5.26.a is always higher than their specified threshold values.

For all sequences of the frames, the tracking errors of all features swing between minimum and maximum limits. The tracking error (RMS) for MFPPF is the preferable and most superior one, as seen in the third column of Table 5.VIII.



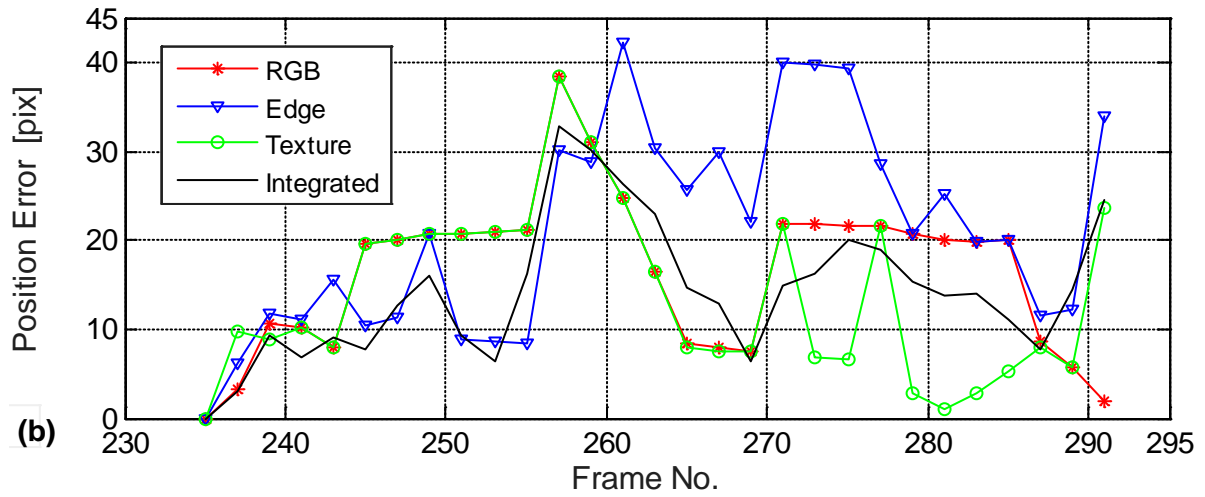


Fig.5.26 Case-VI (a) Individual average similarities, (b) Individual SFPF errors Vs. MFPP error

4) Case-VII: Influence of Full Occlusion

In this scenario, the tracked vehicle for some time (about 70 frames) is obscured by another vehicle traveling along the adjacent road lane, and it becomes completely invisible (Fig.5.27).

The three average similarities of the descriptors are lower than the stated threshold values during complete occlusion between frames #80 and #160, and MFPP runs in "Prediction" mode, as shown in Fig.5.28.a. Starting with frame #150, the "Edge" feature generates significant results, while the color and texture features remain excluded, and MFPP primarily serves as a single feature particle filter based on the "Edge" feature. The fourth column of Table 5.VIII points out that the MFPP has the best RMS tracking accuracy for the entire time compared with SFPF.

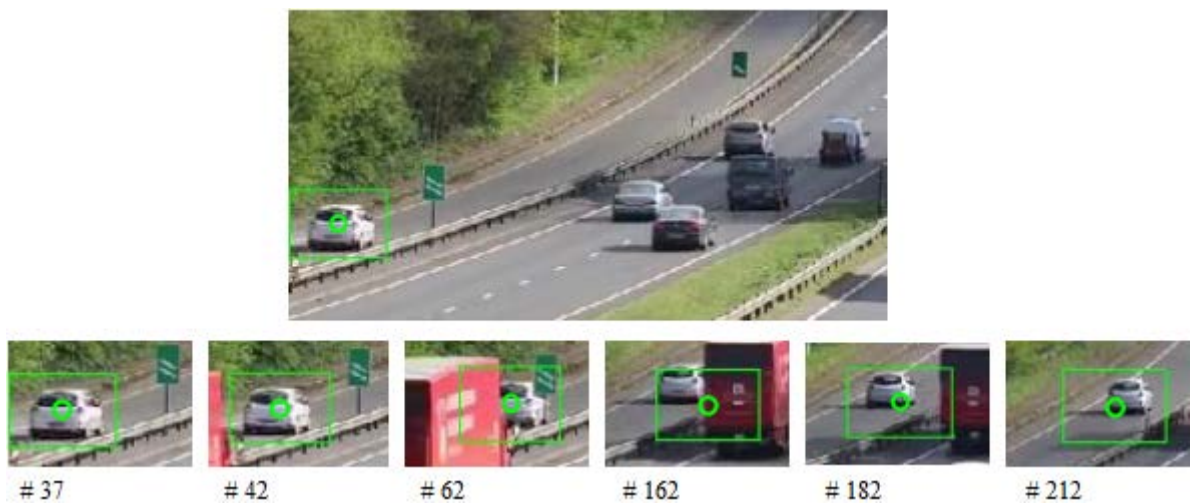


Fig.5.27 Illustration of Case VII

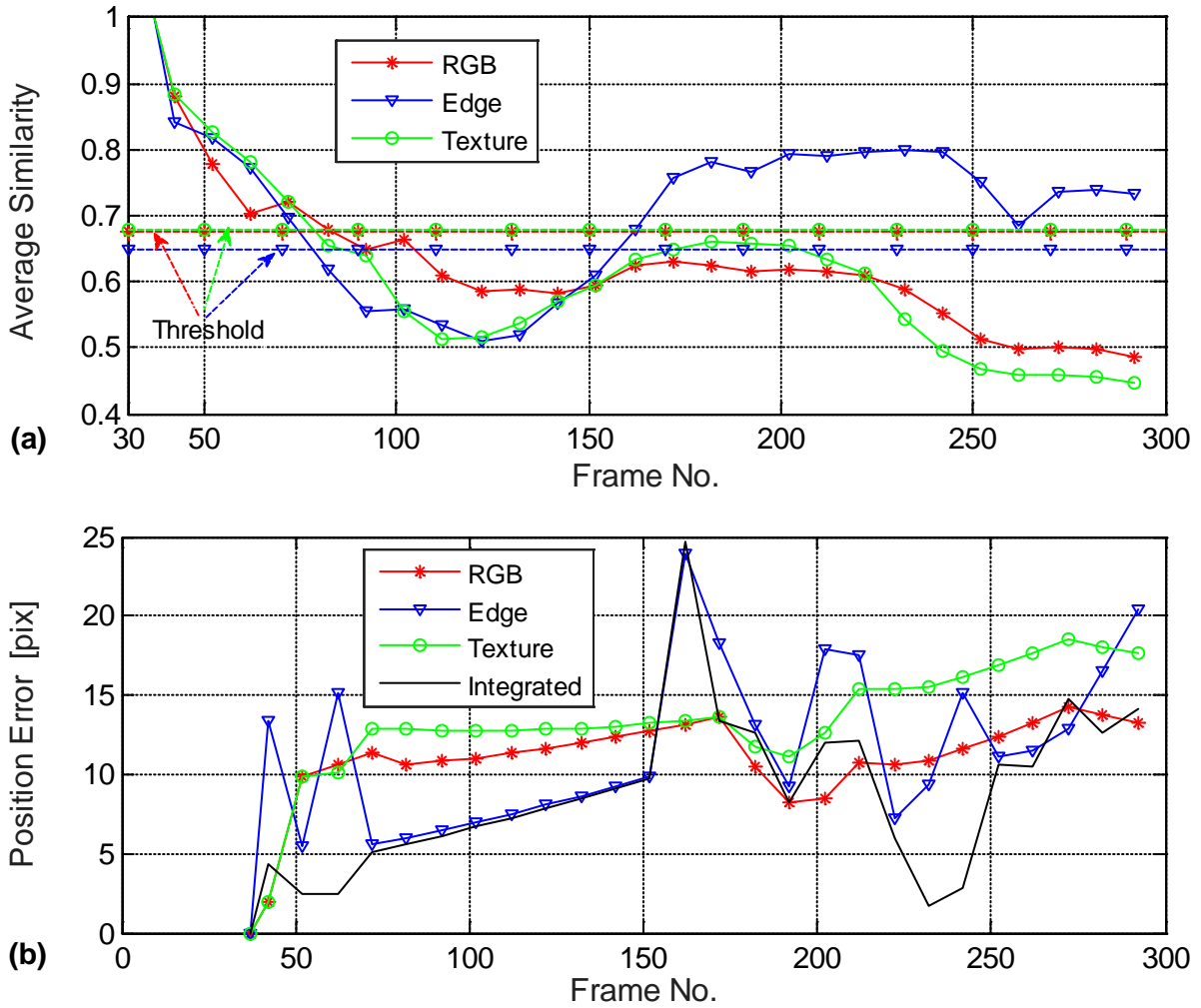


Fig.5.28 Case-VII: (a) Individual average similarities, (b) Individual SFPP errors Vs. MFPP error

5) Case-VIII: Combined Effect of Shadows and Full Occlusion

In the following scenario, the shadows produced by the vehicle itself and trees, partial occlusions, and a brief period of complete occlusion, are combined, as well as gradual zooming out of the size of the vehicle as shown in Fig.5.29. The "edge" feature is the lower responsive to these perturbations, as is seen in Fig.5.30a.

Due to the entire occlusion of the tracked vehicle between intervals #350 and #380, MFPP operates in "Prediction mode". Before and after this period directly, MFPP was only managed by the "Edge" descriptor of SFPP. Outside of this time, the other two features have a nearly equal impact.



Fig.5.29 Illustration of Case VIII

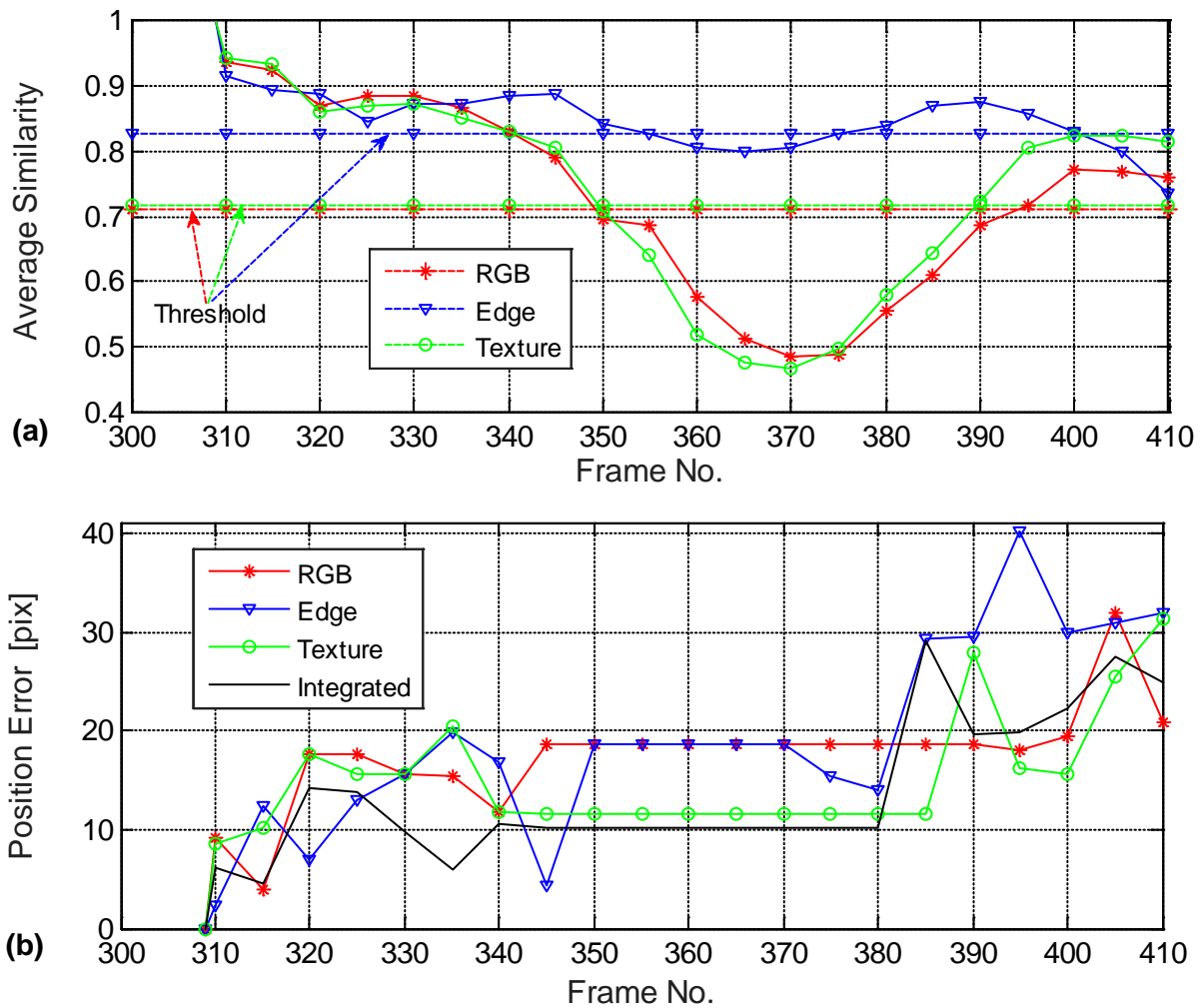


Fig.5.30 Case-VIII: (a) Individual average similarities, (b) Individual SFPP errors Vs. MFPP error

The fifth column of Table 5.VIII shows that the MFPP has a less RMS tracking accuracy compared with SFPP.

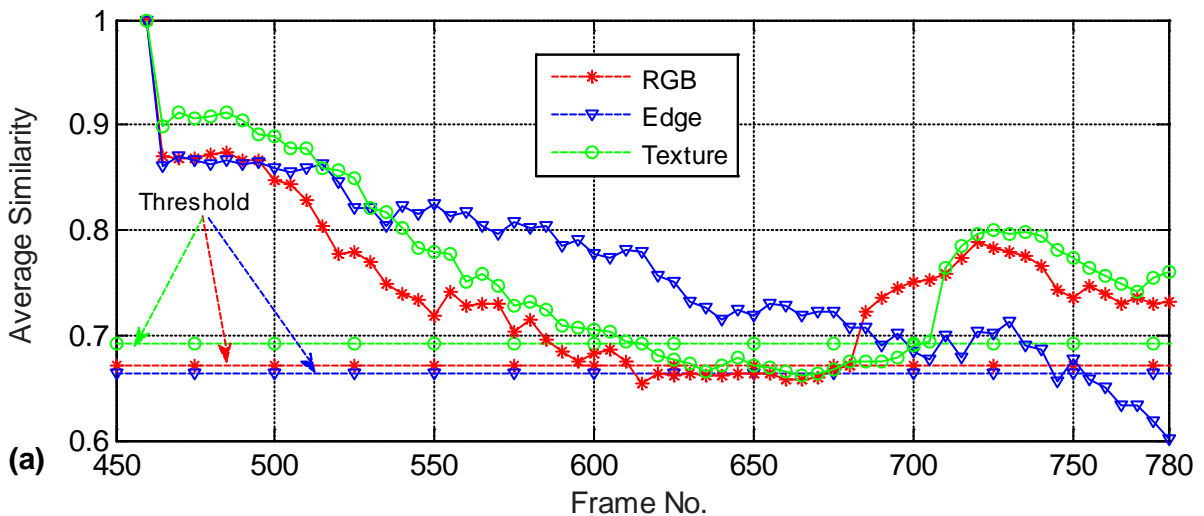
6) Case-IX: Influence of Full Occlusion Accompanied by Maneuvering

This case illustrates the influence of partial and complete occlusion. During the interval of maneuvering, the tracked vehicle is subjected to partial and entire occlusion while making a turning to the left. There is a relatively small interval at which all three features generate valueless estimates in the interval of complete occlusion (Fig.5. 31) (since average similarities are lower than the specified appropriate thresholds).



Fig.5. 31 Illustration of Case IX

Once the occlusion is gone, both "RGB" and "Texture" features become the main reasonable sources of information (Fig.5.32). Table 5.VIII's sixth column demonstrates the superiority of the MFPP technique.



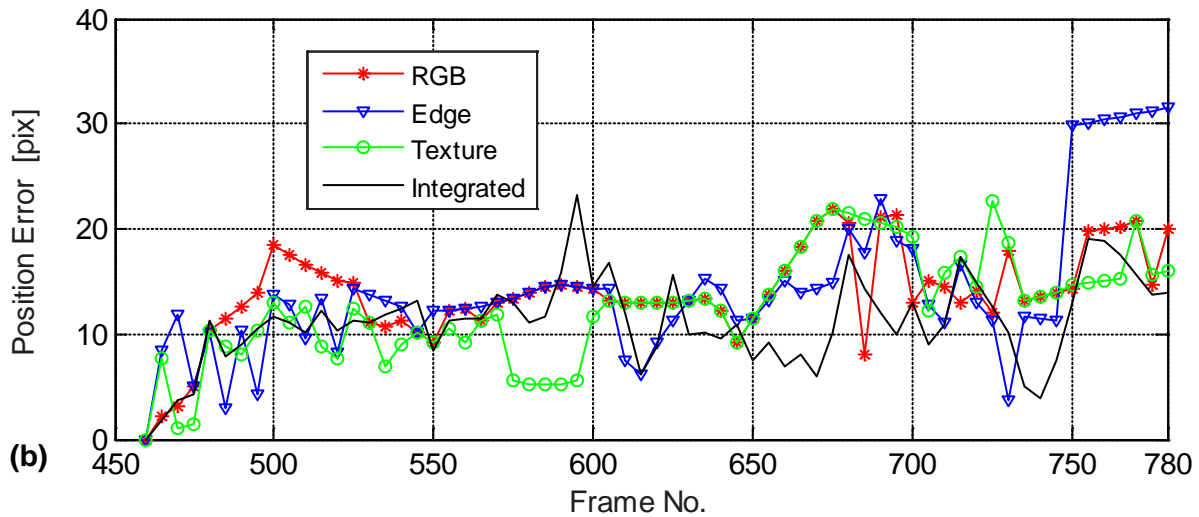


Fig.5.32 Case-IX: (a) Individual average similarities, (b) Individual SFPF errors Vs. MFPP error

Table 5.VIII exhibits the obtained RMS errors that illustrate the SFPF and the MFPP results accuracy for the six real scenarios for different cases and situations. In each of the six scenarios, it is clear that the MFPP has given the lowest RMS tracking error. When the obtained outcomes of individual SFPFs are compared, it becomes clear that no ones of the three features (a single SFPF) could be proclaimed as the best in all conditions in every scenario. There are some periods during the scenario where any one of the SFPF could be the best candidate, while in others; this SFPF might be the worst. The RMS tracking errors produced by a three-feature-based filter (MFPP) which uses all the image's three features are always 10%–25% fewer than the least RMS tracking errors of SFPF.

In general, the results achieved by this MFPP showed advantages that outweigh all other "Single-Feature Particle Filters" (SFPF) and remain agreeable from the aspect of overall computation time.

Table 5.VIII
RMS TRACKING ERRORS (PIXELS)

		Case IV	Case V	Case VI	Case VII	Case VIII	Case IX
SFPF	RGB	36.40	13.50	18.51	11.23	17.81	14.52
	Edge	17.61	10.89	24.01	12.54	21.02	15.77
	Texture	26.35	11.95	16.69	13.66	16.04	13.50
MFPP		15.21	8.38	16.16	9.97	15.11	11.93
MFPP Relative to the best SFPF [%]		86.37	76.95	96.82	88.78	94.20	88.37

To highlight the MFPP algorithm's comparative advantages, two new statistical measures of quality have been included in the real scenarios (Case-IV to Case-IX). Table 5.IX below shows the probability that any of the particular SFPF has had an average similarity greater than a certain

threshold, which has been defined as 80% of the primary average similarities. Regarding the status of MFPP, Table 5.IX's final row depicts the odds that MFPP is not in prediction mode. And it is self-evident that when the SFPPs lose the tracked object, or when all of the SFPPs' outcomes are inadmissible, the MFPP must switch to just a prediction mode, which is less often than any other SFPP.

Table 5.IX

PROBABILITY THAT THE AVERAGE SIMILARITY IS ABOVE ITS THRESHOLD

		Case IV	Case V	Case VI	Case VII	Case VIII	Case IX
SFPP	RGB	88.3 %	100 %	100 %	22.2 %	59.0 %	80.0 %
	Edge	95.3 %	100 %	100 %	70.3 %	77.2 %	89.2 %
	Texture	95.3 %	100 %	100 %	18.5 %	63.6 %	72.3 %
MFPP		100 %	100 %	100 %	73.1 %	85.7 %	100 %

Finally, Table 5.X shows the probabilities that the tracking errors are less than the threshold of 15 pixels. We have embraced this threshold value as a relevant one based on the collection of cases already evaluated, average distances from monitored objects and their sizes, as well as image resolution. From this perspective, one might conclude that MFPP is superior.

Table 5.X

PROBABILITY THAT THE TRACKING ERROR IS LESS THAN 15 PIXELS

		Case IV	Case V	Case VI	Case VII	Case VIII	Case XI
SFPP	RGB	6.9 %	63.4 %	37.9 %	100 %	18.1 %	70.7 %
	Edge	67.4 %	78.0 %	37.9 %	70.3 %	31.8 %	76.9 %
	Texture	23.2 %	78.0 %	55.1 %	66.6 %	59.0 %	72.3 %
MFPP		62.7 %	100 %	62.0 %	96.3 %	72.7 %	84.6 %

Chapter 6

Conclusion

6 Conclusion

Most of the known visual object tracking techniques in the world that intend to the estimation of kinematic parameters of the moving object are met by essential tasks using computer vision. These tracking techniques face many defies such as variable background, occlusion, and others with the increase in the demand for accurate object detecting and tracking. This means that the tracking techniques have to be improved by evolving the existing tracking approaches.

This research introduces the algorithm of video tracking of vehicles based on a particle filter that uses a fusion of three individual filters relying on three features of the concerned image. All three proposed features (color, edge, texture) are represented in the standard form of histograms. The basic assumption was that none of the typical image features can be used alone for all different traffic conditions with different disturbances. Rather than attempt to improve any one of the single features/descriptions, we decided to use the combination of three of them (the color, the edge in the form of orientation of gradients, and texture as gray-level-co-occurrence-matrix) and expressed independently via different probability distributions – histograms.

A Bhattacharyya distance measure is used to specify the particle position that has a maximal similarity of the appropriate histogram compared with the reference histogram, in order to estimate three candidates for the estimated position of the tracked object. The final estimate is calculated from the weighted averaging of three candidates using adaptive weighting factors depending on average similarities in the particles set.

A higher level of adaptability is reached by excluding some of the features when their average similarity measures are below particular threshold values. Individual particle filters based on a single feature have different sensitivities relative to typical disturbances, like variable shadows and reflections, partial and full occlusions, variable background, etc. that affect the tracking process.

Verification of the performance of the proposed multi-feature-particle-filter was initially done by analyzing simple traffic of artificial image sequences, followed by typical real traffic scenarios on roads. In all synthetic image sequence scenarios and in all realistic traffic sequence scenarios, the Integrated PF algorithm was the best, and it was superior.

Whilst this MFPPF was poorer to the best SFPF in different sub-intervals of the tracking sequence, it looked to be superior overall in terms of RMS tracking error throughout the sequence. RMS tracking errors on the whole interval for the artificial sequences have been 21% – 43% less than the least RMS errors of any SFPF, whereas for the real sequences have been from 10 % to 25 %. As it was shown in Tables VII, and VIII, the MFPPF algorithm is also superior relative to any of SFPF because of the less frequent need to go to prediction mode, as well as because of the higher probability that the tracking error is inside the specified tolerance zone.

The computational burden due to the tripling of particle filter calculation times was reduced by adopting a relatively low number of particles (200). Through an adequate selection of minimal reasonable average similarities, the algorithm excludes and disregards some SFPF results if their average similarity related to a particular feature is less than the threshold. If all three features provide inadmissible results (e.g. full occlusion), the algorithm switches to prediction mode and

continues the work together with reinitializing the particle distribution and gradually expanding the area where they are spread. Weighting coefficients that differentiate individual affects of image features are automatically adapted, based on their average similarities in the set of particles. Parameters of the MFPPF algorithm have been constant here for all illustrative cases. Wherever there exists some higher level of prior knowledge about traffic scenario on the particular part of the road, there is a possibility for re-parameterization of MFPPF (thresholds, window size, number of particles, forgetting factor, etc.) in order to adjust to this situation, and to improve the tracking accuracy.

Our future study of this work will be oriented towards improvements of the algorithm relative to the effects of target resizing (zooming) and rotation, as well as to the natural extension in the case of multiple vehicles.

7 Bibliography

- [1] B. Sharma and N. Kaur, "Various Methods for Object Tracking-A Review," *Journal of Computer Engineering (IOSR-JCE)*, vol. 18, p. 3, 2016.
- [2] C. Zhu, "Video object tracking using SIFT and mean shift," Master of Science Master Thesis, Department of Signals and Systems, Chalmers University of Technology, Gothenburg, Sweden, 2011.
- [3] R. K. Rout, "A survey on object detection and tracking algorithms," Master Of Technology, Computer Science and Engineering, National Institute of Technology Rourkela, Rourkela, India, 2013.
- [4] G. S. S. Ritika, "Moving Object Analysis Techniques in Videos-A Review," *IOSR Journal of Computer Engineering*, vol. 1, p. 6, 2012.
- [5] L. Mihaylova, P. Brasnett, N. Canagarajah, and D. Bull, "Object tracking by particle filtering techniques in video sequences," *Advances and challenges in multisensor data and information processing*, vol. 8, p. 9, 2007 2007.
- [6] M. S. A. Meshram and A. Malviya, "Vehicular Traffic Surveillance for Real Time Using Multiple Methodologies," *International Journal of Scientific & Engineering Research*, vol. 4, p. 5, May-2013 2013.
- [7] R. Kachach and J. M. Cañas, "Hybrid three-dimensional and support vector machine approach for automatic vehicle tracking and classification using a single camera," *Journal of Electronic Imaging*, vol. 25, p. 033021, May/Jun 2016 2016.
- [8] S. Srivastava and E. J. Delp III, "Video-based real-time surveillance of vehicles," *Journal of Electronic Imaging*, vol. 22, p. 16, 2013.
- [9] K. K. Ng and E. J. Delp, "New models for real-time tracking using particle filtering," in *Visual Communications and Image Processing 2009*, 2009, p. 72570B.
- [10] B. Coifman, D. Beymer, P. McLauchlan, and J. Malik, "A real-time computer vision system for vehicle tracking and traffic surveillance," *Transportation Research Part C: Emerging Technologies*, vol. 6, p. 18, 1998.
- [11] M. Z. Islam, C.-M. Oh, and C.-W. Lee, "Video based moving object tracking by particle filter," *International Journal of Signal Processing, Image Processing and Pattern*, vol. 2, p. 14, 2009.
- [12] G. Han, S. Dong, N. Sun, J. Liu, K. Du, and X. Li, "Robust local L₂, 1 tracker via red-green-blue color channel fusion," *Journal of Electronic Imaging*, vol. 25, p. 053002, 2016.
- [13] X. Chen, "Automatic vehicle detection and tracking in aerial video," A Doctoral Thesis Doctor of Philosophy of Loughborough University, Computer Science, Loughborough University, 2016.

-
- [14] M. Isard and A. Blake, "Condensation—conditional density propagation for visual tracking," *International journal of computer vision*, vol. 29, pp. 5-28, 1998.
- [15] A. A. Abdulla, S. Graovac, V. Papic, and B. Kovacevic, "Triple-feature-based Particle Filter Algorithm Used in Vehicle Tracking Applications," *Advances in Electrical and Computer Engineering*, vol. 21, pp. 3-14, 2021, DOI: 10.4316/AECE.2021.02001.
- [16] W. Dong, F. Chang, and Z. Zhao, "Visual tracking with multifeature joint sparse representation," *Journal of Electronic Imaging*, vol. 24, p. 013006, 2015.
- [17] P. Brasnett, L. Mihaylova, D. Bull, and N. Canagarajah, "Sequential Monte Carlo tracking by fusing multiple cues in video sequences," *Image and vision computing*, vol. 25, pp. 1217-1227, 2007.
- [18] P. G. Bhat, B. N. Subudhi, T. Veerakumar, V. Laxmi, and M. S. Gaur, "Multi-feature fusion in particle filter framework for visual tracking," *IEEE Sensors Journal*, vol. 20, pp. 2405-2415, 2019.
- [19] S. M. Muddamsetty, D. Sidibé, A. Trémeau, and F. Mériaudeau, "Salient objects detection in dynamic scenes using color and texture features," *Multimedia Tools and Applications*, vol. 77, pp. 5461-5474, 2018.
- [20] A. Ali, A. Jalil, J. Niu, X. Zhao, S. Rathore, J. Ahmed, *et al.*, "Visual object tracking—classical and contemporary approaches," *Frontiers of Computer Science*, vol. 10, pp. 167-188, 2016.
- [21] M. Yazdi and T. Bouwmans, "New trends on moving object detection in video images captured by a moving camera: A survey," *Computer Science Review*, vol. 28, pp. 157-177, 2018.
- [22] M. Z. Islam, M. S. Islam, and M. S. Rana, "Problem Analysis of Multiple Object Tracking System: A Critical Review," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 4, pp. 374-377, 2015.
- [23] T. Chakravorty, "Robust Face Tracking in Video Sequences," Philosophiæ Doctor, Département de Dénie Informatique et Génie Logiciel Université de Montréal, 2017.
- [24] J. Scharcanski, A. B. de Oliveira, P. G. Cavalcanti, and Y. Yari, "A particle-filtering approach for vehicular tracking adaptive to occlusions," *IEEE Transactions on Vehicular Technology*, vol. 60, pp. 381-389, 2010.
- [25] S. You, H. Zhu, M. Li, and Y. Li, "A Review of Visual Trackers and Analysis of its Application to Mobile Robot," *arXiv preprint arXiv:1910.09761*, 2019.
- [26] W. L. Khong, W. Y. Kow, Y. K. Chin, I. Saad, and K. T. K. Teo, "Overlapping vehicle tracking via adaptive particle filter with multiple cues," in *2011 IEEE International Conference on Control System, Computing and Engineering*, 2011, pp. 460-465.
- [27] P. Paizakis, "Region-Based object Detection and Tracking," Doctoral dissertation, Technical University of Crete, 2009.
-

-
- [28] A. F. Yaseen, "A Survey on the Tracking of Object in Sequence Image," *International Journal of Computer Science and Mobile Computing*, vol. 7, pp. 112-120, 2018.
 - [29] P. H. L. Shilpa, Sunitha M.R, "A Survey on Moving Object Detection and Tracking Techniques," *International Journal Of Engineering And Computer Science*, vol. 5, pp. 16376-16382, 2016.
 - [30] R. C. Joshi, M. Joshi, A. G. Singh, and S. Mathur, "Object detection, classification and tracking methods for video surveillance: A review," in *2018 4th International Conference on Computing Communication and Automation (ICCCA)*, 2018, pp. 1-7.
 - [31] R. Verma, "A review of object detection and tracking methods," *International Journal of Advance Engineering and Research Development*, vol. 4, pp. 569-578, 2017.
 - [32] S. V. Kothiya and K. B. Mistree, "A review on real time object tracking in video sequences," in *Electrical, Electronics, Signals, Communication and Optimization (EESCO), 2015 International Conference on*, 2015, pp. 1-4.
 - [33] R. A. Hadi, G. Sulong, and L. E. George, "Vehicle detection and tracking techniques: a concise review," *Signal & Image Processing : An International Journal (SIPIJ)* vol. 5, pp. 1-12, 2014.
 - [34] H. S. A. Karaki, S. A. Alomari, and M. H. Refai, "A comprehensive survey of the vehicle motion detection and tracking methods for aerial surveillance videos," *IJCSNS*, vol. 19, p. 93, 2019.
 - [35] L. Kindberg, "A frame differencing algorithm that allows for small camera movements," Master in Computer Science, School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, Stockholm, Sweden, 2020.
 - [36] S. H. Shaikh, K. Saeed, and N. Chaki, "Moving Object Detection Approaches, Challenges and Object Tracking," in *Moving object detection using background subtraction*, ed: Springer, 2014, pp. 5-14.
 - [37] M. Tiwari and R. Singhai, "A review of detection and tracking of object from image and video sequences," *Int. J. Comput. Intell. Res*, vol. 13, pp. 745-765, 2017.
 - [38] N. Paul, A. Singh, A. Midya, P. P. Roy, and D. P. Dogra, "Moving object detection using modified temporal differencing and local fuzzy thresholding," *The Journal of Supercomputing*, vol. 73, pp. 1120-1139, 2017.
 - [39] A. S. Jalal and V. Singh, "The state-of-the-art in visual object tracking," *Informatica*, vol. 36, 2012.
 - [40] J. H. Park, G. S. Lee, J. S. Kim, S. H. Ryu, and S. H. Lee, "Independent Object Tracking from Video using the Contour Information in HSV Color Space," *Indian Journal of Science and Technology*, vol. 9, 2016.
 - [41] H. Kekre and K. Sonawane, "Performance evaluation of bins approach in YCbCr color space with and without scaling," *International Journal of Soft Computing and Engineering*, vol. 3, pp. 203-210, 2013.
-

-
- [42] C. G. Rafael and E. W. Richard, "Digital image processing second edition," ed: Publishing House of Electronics Industry, 2002, pp. 665-669.
 - [43] L. Mihaylova, P. Brasnett, N. Canagarajah, and D. Bull, "Object tracking by particle filtering techniques in video sequences," *Advances and challenges in multisensor data and information processing*, vol. 8, pp. 260-268, 2007.
 - [44] B. Deori and D. M. Thounaojam, "A survey on moving object tracking in video," *International Journal on Information Theory (IJIT)*, vol. 3, pp. 31-46, 2014.
 - [45] A. D. Worrall, R. F. Marslin, G. D. Sullivan, and K. D. Baker, "Model-based tracking," in *BMVC91*, ed: Springer, 1991, pp. 310-318.
 - [46] D. P. Chau, "Dynamic and robust object tracking for activity recognition," Institut National de Recherche en Informatique et en Automatique (INRIA), 2012.
 - [47] A. I. Comport, É. Marchand, and F. Chaumette, "Robust model-based tracking for robot vision," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, 2004, pp. 692-697.
 - [48] D. Sidibé, "Particle Filters and Applications in Computer Vision," ed, 2011.
 - [49] A. Almeida, J. Almeida, and R. Araújo, "Real-time tracking of moving objects using particle filters," in *Proceedings of the IEEE International Symposium on Industrial Electronics, 2005. ISIE 2005.*, 2005, pp. 1327-1332.
 - [50] A. Mahmoud, "Multi-scale particle filtering for multiple object tracking in video sequences," 2018.
 - [51] A. Nakhmani and A. Tannenbaum, "Particle filtering with region-based matching for tracking of partially occluded and scaled targets," *SIAM journal on imaging sciences*, vol. 4, pp. 220-242, 2011.
 - [52] A. W. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah, "Visual tracking: An experimental survey," *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, pp. 1442-1468, 2013.
 - [53] K. Nummiaro, E. Koller-Meier, and L. Van Gool, "Object tracking with an adaptive color-based particle filter," in *Joint Pattern Recognition Symposium*, 2002, pp. 353-360.
 - [54] G. Chenguang, L. Xianglong, Z. Linfeng, and L. Xiang, "A fast and accurate corner detector based on Harris algorithm," in *2009 Third International Symposium on Intelligent Information Technology Application*, 2009, pp. 49-52.
 - [55] H. Zhou, Y. Yuan, and C. Shi, "Object tracking using SIFT features and mean shift," *Computer vision and image understanding*, vol. 113, pp. 345-352, 2009.
 - [56] H. Shuo, W. Na, and S. Huajun, "Object tracking method based on surf," *AASRI Procedia*, vol. 3, pp. 351-356, 2012.
 - [57] L. Chi Qin and T. Soo Siang, "An efficient method of HOG feature extraction using selective histogram bin and PCA feature reduction," *Advances in Electrical and Computer Engineering*, vol. 16, pp. 101-108, 2016.
-

-
- [58] H. T. Nguyen and A. W. Smeulders, "Robust tracking using foreground-background texture discrimination," *International Journal of Computer Vision*, vol. 69, pp. 277-293, 2006.
 - [59] A. Naeem, "Single and multiple target tracking via hybrid mean shift/particle filter algorithms," University of Nottingham, 2010.
 - [60] I. Eckel, "Particle Filters for Airborne Tracking and Lane-Level Map-Matching of Vehicles," TU München, 2015.
 - [61] D. Koller, J. Weber, and J. Malik, "Towards realtime visual based tracking in cluttered traffic scenes," in *Proceedings of the Intelligent Vehicles' 94 Symposium*, 1994, pp. 201-206.
 - [62] S.-H. Lee, "Real-time camera tracking using a particle filter combined with unscented Kalman filters," *Journal of Electronic Imaging*, vol. 23, p. 013029, 2014.
 - [63] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Transactions on signal processing*, vol. 50, pp. 174-188, 2002.
 - [64] P. Barcellos, V. Gomes, and J. Scharcanski, "Shadow detection in camera-based vehicle detection: survey and analysis," *Journal of Electronic Imaging*, vol. 25, p. 051205, 2016.
 - [65] R. Canals, A. Ganoun, and R. Leconge, "Occlusion-handling for improved particle filtering-based tracking," in *2009 17th European Signal Processing Conference*, 2009, pp. 1107-1111.
 - [66] C. Reta, L. Altamirano, J. A. Gonzalez, and R. Medina-Carnicer, "Three hypothesis algorithm with occlusion reasoning for multiple people tracking," *Journal of Electronic Imaging*, vol. 24, p. 013015, 2015.
 - [67] B. Sugandi, H. Kim, J. K. Tan, and S. Ishikawa, "Object tracking based on color information employing particle filter algorithm," in *Object Tracking*, ed: IntechOpen, 2011, pp. 69-88.
 - [68] M. Jaward, L. Mihaylova, N. Canagarajah, and D. Bull, "Multiple object tracking using particle filters," in *2006 IEEE Aerospace Conference*, 2006, p. 8 pp.
 - [69] H. Liu and F. Sun, "Fusion tracking in color and infrared images using joint sparse representation," *Science China Information Sciences*, vol. 55, pp. 590-599, 2012.
 - [70] P. Kumar, M. J. Brooks, and A. Dick, "Adaptive multiple object tracking using colour and segmentation cues," in *Asian Conference on Computer Vision*, 2007, pp. 853-863.
 - [71] J.-C. Chen and Y.-H. Lin, "Accurate object tracking system by integrating texture and depth cues," *Journal of Electronic Imaging*, vol. 25, p. 023003, 2016.
 - [72] L. Turner and C. Sherlock. (2013). *An introduction to particle filtering*. Available: <https://www.coursehero.com/file/40160331/PartileFilteringpdf/>
 - [73] D. Doshi and M. Chankaya, "Particle Filter Based State Estimation of Power System," in *2017 International Conference on Recent Trends in Electrical, Electronics and Computing Technologies (ICRTEECT)*, 2017, pp. 77-82.
 - [74] D. Simon, "Optimal state estimation: Kalman, H infinity, and nonlinear approaches," ed: John Wiley & Sons, 2006, pp. 461-483.
-

-
- [75] N. J. Gordon, D. J. Salmond, and A. F. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," in *IEE proceedings F (radar and signal processing)*, 1993, pp. 107-113 DOI: 10.1049/ip-f-2.1993.0015.
 - [76] X. Lu, T. Izumi, L. Teng, and L. Wang, "Particle filter vehicle tracking based on surf feature matching," *IEEJ Journal of Industry Applications*, vol. 3, pp. 182-191, 2014.
 - [77] S. Kamijo, K. Ikeuchi, and M. Sakauchi, "Vehicle tracking in low-angle and front-view images based on spatio-temporal markov random field model," in *8th World Congress on ITS, Sydney Oct*, 2001, pp. 1-12.
 - [78] N. Easwar and J. Shah, "Object Tracking using Particle Filter," ed, 2003.
 - [79] Y. Dai and B. Liu, "Robust video object tracking using particle filter with likelihood based feature fusion and adaptive template updating," *arXiv preprint arXiv:1509.08182*, 2015.
 - [80] D. J. Bora, A. K. Gupta, and F. A. Khan, "Comparing the performance of L* A* B* and HSV color spaces with respect to color image segmentation," *arXiv preprint arXiv:1506.01472*, 2015.
 - [81] D. Hema and D. S. Kannan, "Interactive color image segmentation using HSV color space," *Science and Technology Journal*, 2020.
 - [82] B. Sugandi, H. Kim, J. K. Tan, and S. Ishikawa, "A moving object tracking based on color information employing a particle filter algorithm," *Artificial Life and Robotics*, vol. 14, pp. 39-42, 2009.
 - [83] F. Bashar, A. Khan, F. Ahmed, and H. Kabir, "Face recognition using similarity pattern of image directional edge response," *Advances in Electrical and Computer Engineering*, vol. 14, pp. 69-77, 2014.
 - [84] L. Shapiro and G. Stockman. (2000). *computer vision: Mar 2000, Linda Shapiro George stockman*.
 - [85] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern recognition*, vol. 29, pp. 51-59, 1996.
 - [86] A. Hadid, "The local binary pattern approach and its applications to face analysis," in *2008 First Workshops on Image Processing Theory, Tools and Applications*, 2008, pp. 1-9.
 - [87] M. Robert, K. Shanmugham, and D. Its'hak, "Textural features for image classification," *IEEE Transactions on Systems Man and Cybernetics*, vol. SMC-3, pp. 610-621, 1973 DOI: 10.1109/tsmc.1973.4309314.
 - [88] C. N. Rao, S. S. Sastry, K. Mallika, H. S. Tiong, and K. Mahalakshmi, "Co-occurrence matrix and its statistical features as an approach for identification of phase transitions of mesogens," *Int. J. Innov. Res. Sci. Eng. Technol*, vol. 2, pp. 4531-4538, 2013.
 - [89] E. Maggio, F. Smerladi, and A. Cavallaro, "Adaptive multifeature tracking in a particle filtering framework," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, pp. 1348-1359, 2007.
-

-
- [90] F. Gustafsson, "Particle filter theory and practice with positioning applications," *IEEE Aerospace and Electronic Systems Magazine*, vol. 25, pp. 53-82, 2010.
- [91] P. A. Brasnett, L. Mihaylova, N. Canagarajah, and D. Bull, "Particle filtering with multiple cues for object tracking in video sequences," in *Image and Video Communications and Processing 2005*, 2005, pp. 430-441.
- [92] J. D. Hol. (2004). *Resampling in particle filters*. Available: <http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-2366>
- [93] A. A. Abdulla, A. Rodic, and S. Graovac, "Vehicle Collision Avoidance in a Dynamic Road Traffic Scenario," in *5th International Conference IcETran 2018*, Palić, Serbia, 2018, pp. 1022-1026.
- [94] A. A. Abdulla and S. Graovac, "Comparative analysis of the usage of different image descriptors in object's video tracking," in *6th International Conference IcETran 2019*, Silver Lake, Serbia, 2019, pp. 174-179.
- [95] F. B. Vidal and V. H. C. Alcalde, *Object Visual Tracking Using Window-Matching Techniques and Kalman Filtering*: INTECH Open Access Publisher, 2010.

Biography

Abdalgail Alsagair Mohamed Abdulla was born on August 13, 1962, in Ghdamis–Libya. He finished high school in Ghdamis in 1981/1982, and he graduated from Aeronautical Engineering the Faculty of Engineering-Aeronautical Engineering at the University of Tripoli in Libya in 1986/87.

From 1989 to 1996, he has been worked at “Research and Development center” (R&D center) in Tripoli–Libya as an aeronautical engineer. He participated in the following projects: Aerodynamics calculation for S2S missile, designing of Turbo pump of S2S missile engine, Design of 2-degrees simulation program in Fortran language, autopilot and Guidance design for S2S missile.

The author has he enrolled the Master studies in 1997/98, in the department of signal and system at the School of Electrical Engineering, University of Belgrade, he graduated with an overall average score of 8.92, where he defended his Master's thesis on January 13, 2000, entitled “ Digital Autopilot Design for Control of Normal Acceleration” under the mentorship of Prof. Dr. M. Stojić.

Then in the period from 2000 to 2011 at the same R&D center as a research assistant for the development of Autopilot design and guidance system for S2S Missile. Also, he was teaching as a part-time assistant lecturer at colleges and high institutes in Tripoli–Libya from 2007 to 2011. Since 2011–present he has been a lecturer in the College of Civil Aviation and Meteorology, Esbia - Tripoli- Libya.

Later Mr. Abdulla enrolled in doctoral studies at the Department of signal and system School of Electrical Engineering, Belgrade University in 2015/2016, he passed all the required exams.

The author has published 4 scientific papers in the field of object tracking using computer vision.

List of publications

- Ivana Vujanović, Abdalgail Alsagair Abdulla and Stevica Graovac ,“An Image Processing Based Motion Tracker/Estimator for Traffic Control Purposes”,4th Ic Etran Etran Conference on Electrical, Electronics and Computing Engineering, Kladovo, Serbia, pp. AUI1.2.1-5 , 2017, ISBN 978-86-7466-692-0.
- A. A. Abdulla, A. Rodić, and S.Graovac, “ Vehicle Collision Avoidance in a Dynamic Road Traffic Scenario”, in *5th International Conference* on Electrical, Electronics and Computing Engineering, *IcETRAN 2018*, Palić, Serbia, pp. 1022-1026, 2018, ISBN 978-86-7466-752-1.
- A. A. Abdulla and S. Graovac, “ Comparative analysis of the usage of different image descriptors in object’s video tracking”, in *6th International Conference* on Electrical, Electronics and Computing Engineering, *IcETRAN 2019*, Silver Lake, Serbia, pp. 174-179, 2019, ISBN 978-86-7466-785-9.
- A. A. Abdulla, S. Graovac, V. Papic, and B. Kovacevic, "Triple-feature-based Particle Filter Algorithm Used in Vehicle Tracking Applications," *Advances in Electrical and Computer Engineering*, vol. 21, pp. 3-14, 2021, DOI: 10.4316/AECE.2021.02001.

Изјава о ауторству

Име и презиме аутора Абдалгалил Абдула (Abdalgaliil Abdulla)

Број индекса 2015/5048

Изјављујем

да је докторска дисертација под насловом

One Particle Filter Based Algorithm of Tracking of a Moving Object in the Sequence of Images

(Алгоритам праћења покретних објеката у секвенци слика применом честичног филтра)

- резултат сопственог истраживачког рада;
- да дисертација у целини ни у деловима није била предложена за стицање друге дипломе према студијским програмима других високошколских установа;
- да су резултати коректно наведени и
- да нисам кршио/ла ауторска права и користио/ла интелектуалну својину других лица.

У Београду, 11.04.2022

Потпис аутора



Изјава о истоветности штампане и електронске верзије докторског рада

Име и презиме аутора Абдалгалил Абдула (Abdalgali Abdulla)

Број индекса 2015/5048

Студијски програм Докторске академске студије

Наслов рада One Particle Filter Based Algorithm of Tracking of a Moving Object in the Sequence of Images

(Алгоритам праћења покретних објеката у секвенци слика применом честичног филтра)

Ментор Проф. др Бранко Ковачевић, професор меритус

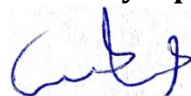
Изјављујем да је штампана верзија мог докторског рада истоветна електронској верзији коју сам предао/ла ради похрањивања у **Дигиталном репозиторијуму Универзитета у Београду**.

Дозвољавам да се објаве моји лични подаци везани за добијање академског назива доктора наука, као што су име и презиме, година и место рођења и датум одбране рада.

Ови лични подаци могу се објавити на мрежним страницама дигиталне библиотеке, у електронском каталогу и у публикацијама Универзитета у Београду.

У Београду, 11.04.2022

Потпис аутора



Изјава о коришћењу

Овлашћујем Универзитетску библиотеку „Светозар Марковић“ да у Дигитални репозиторијум Универзитета у Београду унесе моју докторску дисертацију под насловом:

One Particle Filter Based Algorithm of Tracking of a Moving Object in the Sequence of Images
(Алгоритам праћења покретних објеката у секвенци слика применом честичног филтра)

која је моје ауторско дело.

Дисертацију са свим прилозима предао/ла сам у електронском формату погодном за трајно архивирање.

Моју докторску дисертацију похрањену у Дигиталном репозиторијуму Универзитета у Београду и доступну у отвореном приступу могу да користе сви који поштују одредбе садржане у одабраном типу лиценце Креативне заједнице (Creative Commons) за коју сам се одлучио/ла.

- 1. Ауторство (CC BY)
- 2. Ауторство – некомерцијално (CC BY-NC)
- 3. Ауторство – некомерцијално – без прерада (CC BY-NC-ND)
- 4. Ауторство – некомерцијално – делити под истим условима (CC BY-NC-SA)
- 5. Ауторство – без прерада (CC BY-ND)
- 6. Ауторство – делити под истим условима (CC BY-SA)

(Молимо да заокружите само једну од шест понуђених лиценци.
Кратак опис лиценци је саставни део ове изјаве).

У Београду, 11.04.2022

Потпис аутора

