Vision Based Multi-Target Tracking

Pedro Ribeiro da Silva
Instituto Superior Técnico / UTL, Lisbon, Portugal

ABSTRACT

The interest in automatic surveillance technologies has been increasing in recent years. These technologies can help human operators to detect more events and to deal with larger number of cameras.

In this work a multi-target tracking algorithm, based in a particle filter and the kernel-based tracking algorithm, is proposed. The data association problem inherent to the multi-target scenario is handled using a Global Nearest Neighbors approach (GNN). Furthermore a heuristic scheduling algorithm is also developed, to coordinate a set of pan-tilt-zoom (PTZ) cameras. The proposed tracking methodology and the scheduling algorithm are then combined to produce a complete system able to track targets in delimited areas.

In order to analyze the performance of the tracking algorithm it is compared with other tracking algorithms in a scenario of multiple similar targets. The full system is tested in a virtual setup where several targets enter and leave the tracking area.

Keywords: Multi-target tracking, Particle filter, Visual object detection, Kernel based tracking, Pan-tilt-zoom camera, Camera scheduling, Mixture of Gaussians, Data association.

I. INTRODUCTION

Computer vision systems are ubiquitous nowadays. Visual automatic surveillance systems are of particular interest for our work. These systems require tracking and visual object detection methodologies, to detect and follow targets, and need to be robust specially when used outdoors. Furthermore the reduction in the price of sensors led to an increased interest in the study and implementation of sensor networks.

The above facts motivated this dissertation whose objective is the detection and tracking of multiple targets using a set of pan-tilt-zoom (PTZ) cameras. A special focus is placed in the development of the multi-target tracking methodology with a simple approach chosen for the camera scheduling algorithm. This kind of system has several real world applications like the monitoring of vehicles in highways or in parking lots.

A. Related Work

The problem of visual tracking has been widely studied with several articles and books describing the subject. It can be divided in a component of image processing, necessary to extract measurements for the targets state, and another of filtering where a tracking algorithm is employed. In the case that more than one target is being tracked then an additional problem, called data association, arises. This has to due with the fact that the observations have now to be associated to one of the targets.

In addition to the problem of visual tracking, this work also encompasses the subject of sensor management. It deals with the scheduling of tasks, that a set of agents can perform, in order to achieve a certain goal. In the context of tracking the goal can be to manage the behavior of a set of cameras or radars in order to decrease the tracking error.

The following two sections present an overview of the literature associated to the tracking and the sensor management problems.

B. Tracking

One of the most well known and widely used tracking filters is the Kalman filter [21], originally proposed in 1960 by Rudolph Kalman. This Bayesian tracker can be applied to linear systems with additive Gaussian noise, for which it provides an optimal estimator of the system state. This, together with the fact that it can be easily implemented, led to its great success in several areas of science [8]. For example in [37] a Kalman filter is used to track manned maneuverable vehicles, while in [42] the Kalman filter is applied to the prediction of spot interest rates (the interest rates associated to loans). However the fact that the Kalman filter can only be applied to linear systems restraints its range of applications. This led to the development of adaptations so that it could be applied to non-linear systems. One those examples is the Extended Kalman Filter (EKF) [16]. However the noise is still considered to be additive and Gaussian. This filter employs a linearization of the non-linear model and then applies the standard Kalman filter equations to the resulting system. Due to the linearization, the EKF is no longer optimal. However it still provides accurate enough results since a large set of articles using it have been published. For example in [24] the EKF is used to track the speed of an induction motor, and some years later in [34] the EKF is used to estimate a set of parameters describing the state of a battery, with the objective of using the estimated data in a battery management system. One year later a work is published [44] where the EKF is used to estimate the traffic state of a freeway. More recently the work [29] studied the benefits of using a EKF in the improvement of an unmanned aircraft autopilot system, by providing better estimations of a series of state variables that this system uses to take decisions.

Another variant of the Kalman filter is the Unscented Kalman Filter (UKF) [20] proposed by Julier and Uhlman in 1997, which like the EKF is also applicable to non-linear systems. The core idea of the UKF is the use of a sampling technique called unscented transform [43] which does not require any system linearization and is able to provide better approximations for the system state. However the assumption of additive Gaussian noise is still assumed. Some applications of these tracker are the work of Stenger *et al.* [40], in which an UKF is used to track the pose of a human hand, and the more recent work of Kim *et al.* [23] where the unscented kalman filter is proposed to improve the positioning accuracy of automatic guided vehicles using laser navigation systems.

A different kind of Bayesian filter is the particle filter, which uses Monte Carlo techniques to provide estimates of the targets state [16]. One of the advantages over the EKF and UKF is that this tracker can be applied to non-linear non-Gaussian systems. The main idea of this filter is to approximate the *posterior* distribution of the target state given the observations, using a set of weighted samples. A state estimate can then be obtained, typically, by computing the expected value of the *posterior* distribution. As demonstrated in [1] the particle filter can outperform the EKF, given that an adequate number of samples is chosen. However the particle filter is more computationally burdensome which can lead to the EKF being preferred in some situations [1]. In the work [4] the UKF, the EKF and the particle filter are compared in the task of people tracking using a mobile robot. Like in [1] the particle filter also produces better estimates than the EKF, however in the case of the UKF the root mean squared errors obtained are very similar, with the particle filter only providing slightly better results. Another example of the application of the UKF in a typical tracking scenario, is published by Kambhampati *et al.* in [22] where the data received from a radar is used to estimate the target position and velocity using a particle filter. In the work of Montesano *et al.* [31] a combination of a particle filter and a EKF is used in the tracking of a pair of robots. In this work the tracks are initially generated using a particle filter, with the distribution, approximated by the particles, being periodically tested to check if a Gaussian can fit it. In that case the tracking is switched to a EKF. This solution showed good results for the application concerned.

The original concept of the particle filter has also led to several variants. One of them is called Rao - Blackwellized Particle Filter (RBPF) [15], which tries to explore the structure of the system model in order to factorize the *posterior* distribution into two probability distributions. The state variables involved in one of these distributions can only have a linear Gaussian structure so that the Kalman filter can be applied. The other distribution has the remaining non-linear variables and is approximated using a particle filter. The RBPF can be particularly useful to deal with high-dimensional problems, which the particle filter has difficulty to handle , since the number of particles necessary to adequately model the *posterior* distribution scales exponentially with the state space size [38]. In the work [27] a RBPF is used in the tracking of a ball and implemented in a four-legged robot for the context of robotic soccer.

Another variant of the particle filter is the hybrid tracker of Maggio and Cavallaro [28]. This variant uses the kernel-based tracker of Comaniciu *et al.* [9] to guide the particles states to values that have higher weights. This way more particles are placed near the peaks of the *posterior* distribution. The kernel-based tracker of Comaniciu *et al.* is a non-bayesian method directed to the tracking of objects in a video using a reference histogram describing the target in the image.

Up until now the methods mentioned do not explicitly handle the scenario of multi-target tracking, which has mentioned before, generates a problem of data association. One of the most known algorithms to deal with this type of problem is the Multi Hypothesis Tracker (MHT) of Reid [35]. The main idea of this algorithm is to compute all the possible joint measurement-to-target hypothesis, each one with a probability assigned, that tells the likelihood of that association. At each iteration these hypothesis are propagated into the future, generating an hypothesis tree, with the goal of acquiring more information that could help solve any association conflicts, like the ones arising when a measurement is associated to two targets. This method has shown to provide good results has mentioned in [6]. The major disadvantage is however the computational burdensome necessary to maintain the hypothesis tree. Several techniques are used to deal with this problem like pruning the tree according to certain rules and limiting its depth.

Another approach to the data association problem is the Joint-Probabilistic Data Association Filter (JPDAF) [12] which is based in a Bayesian framework. As mentioned in [19] this method can be considered has a special case of the MHT, where the hypothesis are not propagated into the future. Instead at each time step infeasible hypothesis are discarded (like the ones that associate two targets to the same measurement), with the remaining hypothesis receiving a likelihood value and combined in the *posterior* update [41]. The JPDAF was initially designed to deal with linear Gaussian systems however in [41] a complete JPDAF framework for general non-linear non-Gaussian systems is proposed, using a Monte Carlo based approach. This method, although less computation expensive than the MHT, is still computationally burdensome.

A simpler and less computational approach to data association is the Global Nearest Neighbors (GNN) method [33]. Instead of considering the joint measurement-to-target association hypothesis like is done in the MHT and JPDAF, the main idea is to compute the cost of assigning each measurement to each of the targets individually. These computed costs are then used to form a cost matrix at which the Hungarian algorithm (or other method to solve assignment problems) is applied ot obtain an assignment of measurements to targets. Due to the simplicity of this method it is advantageous in situations where the number of observations is high, like in scenarios of elevated clutter.

C. Sensor Management

The problem of sensor management can be seen as a decision process, where each agent is defined by a state for which a set of actions are allowed. This agent uses then the set of observations it receives to define an action policy [17]. A typical assumption is to consider that the decision process is Markovian meaning that the current states of the agents only depend on the previous states. This type of processes are called in the literature Markov Decision Processes (MDP) or Partially Observable Markov Decision Processes (POMDP) when there are underlying state variables that cannot be directly observed [17]. Formulating the sensor management problem this way allows to use well studied techniques from decision theory as described in [17]. In the context of decision theory and sensor management there is a particular type of decision processes that are commonly used in sensor management. They are called Multi-armed Bandit (MAB) decision processes and in its original formulation they model the problem of a controller that can select one, among a series of machines, to operate. When at a given time a machine is selected its state is allowed to change and a reward is received, with all the remaining machines staying frozen [18]. The state of each machine is assumed to be independent of the others. This type of decision processes admit an optimal policy designated Gittins Index [18], [13] which, in the analogy of the controller and machines, consists in computing for each machine a value, the Gittins index, and then selecting at each iteration the machine with the highest index.

The original formulation for the MAB problem however restraints its applicability to a little number of practical applications, mainly because it forces the not selected machines to remain stationary. For example in a setup of a camera tracking several vehicles this condition will not hold since when the camera selects a vehicle to track, the state (which can be their position) of the other ones most likely will change. This situation and others that appear in sensor management led to a wide variety of MAB variants. The most known ones are entitled Arm-acquiring bandits, Multiple Plays and the Restless Bandits [17]. In the case of the first, as the name suggests, arrival of new machines is allowed. This could be useful for a tracking scenario where the number of targets can change over time. For this variant the Gittins index is still optimal [17]. In the case of the Multiple Plays the controller can now select more than one machine to operate at each iteration. The Gittins index is no longer optimal for this variant. However an optimal scheduling policy exists has mentioned in [17]. Finally the last variant mentioned, the Restless Bandits, extends the original MAB formulation to the situation in which the state of the machines that are not operated is allowed to change. This is the most complex variant mentioned, and no general optimal policy is known. There are however particular situations that allow the restless bandits to have an optimal policy as stated in [17]. To give some examples of applications of the MAB to concrete problems, consider the work of Ahmad *et al.* [2] which formulates the problem of a user that can select at each iteration k of n communication channels, as a restless bandits with multiple plays. Another example is the work of Ny *et al.* [32] which uses the restless bandits framework to model the problem of scheduling the visit of a set of vehicles by a series of locations.

The sensor management problem can also be seen in the context of information theory [17]. The type of policies that result from the application of methods fitting in this category are typically labeled myopic policies. This has to due with the fact that MDP/POMDP and MAB approaches involve a multi-stage policy in which each available action has to be evaluated in terms of its impact on the potential future rewards. Information theory driven policies on the other hand are typically interested only in the immediate or near-immediate reward [17], and are characterized by the definition of measures used to define the value of a certain action at a given time step. The action with the best measure is the one selected. Examples of these kind of measures are the Kullback-Leibler divergence and the Rényi entropy [17]. For instance the work [25] uses a information theoretic approach to handle the problem of sensor management in large sensor networks, with the Rényi entropy used to measure the value of an action.

Finally one other approach to sensor management is the use of heuristics. An example of this is the work of Tiago Castanheira *et al.* [7] where a set of PTZ cameras are allocated to a series of moving targets using a queue. At each iteration the targets to be followed are pulled from the queue. A more complex example is the work of Starzyk and Qureshi [39] which define the behavior followed by a set of PTZ cameras using a state-machine.

D. Problem Formulation

The problem of visual tracking comprises a set of challenges. The detection of the targets in image sequences, requires dealing with illumination changes, and with geometric imaging changes that the targets show due to their motion. These types of problems may be handled with techniques that are invariant to the aforementioned changes like the SIFT algorithm. Occlusions may happen due to background objects or another targets. The most challenging ones are those in which the target is completely occluded for a considerable amount of time, where it is usually necessary to use a tracking filter combined with a motion model, in order to estimate the most probable trajectory taken by the target. When more than one target is being tracked at the same time then an additional problem, called data association, arises due to the necessity of associating measurements to the adequate targets.

In what concerns the problem of camera scheduling, it is necessary to deal with the situation where the number of cameras is less than the number of targets. This implies assigning more than one target to a camera. Another challenge that arises in camera scheduling is the management of the camera behaviors so that they maximize the field of view and at the same time search effectively for new targets

E. Paper Structure

Section 1 introduces the problem to approach in the thesis, in particular presents a short discussion on the state of the art of some algorithms of tracking and camera scheduling. Section 2 describes briefly the multi-target tracking algorithm developed and explains the camera scheduling algorithm. Section 3 provides a detailed description of the multi-target tracking algorithm. Section 4 presents the experiments and respective results obtained. Section 5 summarizes the work performed and highlights the main achievements in this work. Moreover, this chapter proposes further work to extend the activities described in this document.

II. Multi-Camera Tracking System

The system developed in this project has essentially two main components, namely (i) events detection and tracking and (ii) the camera scheduling algorithm (CSA). Our main effort is placed in the detection and tracking component. In particular we are interested in dealing with multiple similar targets, a common situation in many applications. This aspect is studied in detail just in the next chapter. In what concerns the CSA it is designed to deal with multiple PTZ cameras. Our approach to this aspect is detailed in this chapter.

III. Multi-Camera Scheduling

The multi-camera scheduling algorithm developed in this project is targeted for closed spaces where the number and location of entrances and exits of targets is known. Furthermore it is assumed that there are sensors (i.e. infrared or motion sensors) that are activated when a target enters the area being observed. The activation of these sensors triggers an associated static camera placed at each entrance, which then takes a picture used to build the target visual model. The number of targets is considered to be known at each time step and it is assumed that they are moving in a plane, designated tracking plane, where their z coordinate is constant. To follow the targets a set of PTZ cameras is placed inside the tracking area. A suitable choice of the number and location of the PTZ cameras can improve the tracking performance, however this is not analyzed in this work.

The need for a camera scheduling algorithm is associated with the fact that the cameras are a limited resource, which can, at some point in time, be inferior to the quantity of targets. The scheduling algorithm is therefore responsible for making decisions of where and when to move the PTZ cameras, and to inform the tracking algorithm the camera frames it should look in order to search for the targets. In order to adapt the target information there is also a procedure responsible for removing targets that left the tracking area, and adding new detected targets.

This algorithm receives several parameters in the moment the system is initialized. These are: (i) intrinsic and initial extrinsic parameters of the cameras; (ii) cameras base rotation; (iii) surveillance area; (iv) field of view (FOV) margin (v) pan and tilt maximum angles and maximum speed. The parameter (iii) allows to restraint the area a camera may observe and corresponds to a 2×2 matrix whose columns are the coordinates of two rectangle vertices (although for this project only rectangular areas were needed it easy to extend the system for other types of polygons). This can be particularly useful in situations where a camera is unable to observe part of the tracking area due to occluding objects.

In what concerns the parameter (iv) it is used to virtually decrease the FOV, and corresponds to a vector with two elements, one for the image width and another for the height, which indicates a margin in pixels relatively to the image boundaries. This virtually decreases the dimension of the image captured by the camera and is used to handle problems arising from situations of targets that are only partially projected in the image.

Beyond these user defined parameters, one of three states is assigned to each camera. These states are updated at each iteration by the algorithm according to the finite state machine shown in figure 1. If the camera is in state 0 then there is no

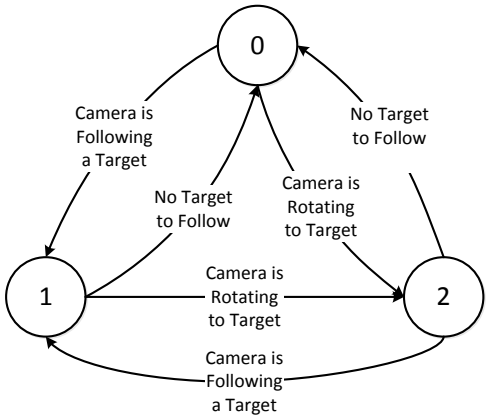


Fig. 1. State machine of the cameras states

target to follow and therefore the camera is stopped. When a camera is assigned to a target, it may need to rotate, depending on the relative position of the target, in order to have it in its FOV. When for a given camera the target is already in the camera FOV, its state is changed to 1. If this condition is not met then the camera state is set to 2, and while in this state no other target can be assigned to it. This restriction has the goal of avoiding some situations in which the camera starts to oscillate between two targets without ever observing one.

The objective of the CSA is then to assign the valid cameras contained in the set V_t to a target. This is an assignment problem which can be solved using the Hungarian algorithm [26]. It is however necessary to define the cost matrix $\mathcal{C}$ for the problem to be complete. In this scheduling algorithm the matrix $\mathcal{C}$ has two different definitions. While there are less targets than cameras the cost matrix entries $\mathcal{C}_{(i,j)}$, are equal to the distance between the camera associated to the row index i and the target of the column index j. In other words $\mathcal{C}_{(i,j)} = \left\| \boldsymbol{P}_{C_{\{i\}}} - \boldsymbol{P}_{T_{t,\{j\}}} \right\|$, with $\boldsymbol{P}_{C_{\{i\}}}, \boldsymbol{P}_{T_{t,\{j\}}}$ denoting the position of camera $C_{\{i\}}$ and target $T_{t,\{j\}}$ respectively. This holds the idea that each camera should be assigned to the closest target, which is based on the fact that as the target

moves away from the camera, the number of pixels it occupies in the image tends to decrease. Therefore as the distance increases the signal-to-noise ratio decreases [10], becoming harder to track the target. The cameras left without a target are assigned to their closest target.

On the other hand if the number of targets is higher than the number of cameras, the aforementioned definition for the cost matrix can give poor results, and eventually lead to track lost, since there are targets which are not observed. In this case the idea is to associate to each camera the target that maximizes the number of targets in its FOV, therefore avoiding the absence of target observations. In other words the cost matrix entries $\mathcal{C}_{(i,j)}$ are the number of targets whose projected coordinates are inside the image boundaries of camera i, when target j is in its center of view.

After the execution of the Hungarian algorithm the resulting assignment, together with the current state estimate, is used to generate the commands that are sent to move the cameras so that they place the tracking target at the center of the image. The system then moves to the next iteration, which may or may not be enough time for the cameras to complete the whole movement assuming that their velocity is finite. Therefore a query is sent to the cameras in order to retrieve their current pose, which is then used to update the cameras states according to the state machine of figure 1, defining this way the camera states for the next iteration.

The algorithm then proceeds to the last step, where it computes for each target the cameras that have it in their FOV. The resulting camera indices are stored in a vector and assigned to the respective target. Finally this information is sent to the tracking algorithm so that it knows the camera frames it can use to acquire observations of a given target.

IV. Multi-Target Tracking

The multi-target tracking algorithm detailed in this section is inspired by the work of Maggio and Cavallaro [28] which developed an hybrid single target tracking algorithm that combines a particle filter and the kernel-based tracking (KBT) method of Comaniciu et al. [9]. To improve this work the tracking algorithm proposed, presents a new way to adapt the scale of the target region of interest (ROI) by using additional geometric information. Furthermore it also handles the problem of multi-target tracking by adding a step of data association using a Global Nearest Neighbor (GNN) approach.

The multi-target tracking algorithm proposed, is entitled Multi-target Hybrid Particle Filter (MHPF), and can be divided in several steps. These are described in the following sections.

A. Prior Update

Like in the work of Maggio and Cavallaro [28] the MHPF associates to each target k an individual particle filter. However the estimated state vector $\hat{\boldsymbol{x}}_t^k$ chosen has four dimensions, which are the target position $\hat{x}_t^k, \hat{y}_t^k$ in the tracking plane, assumed to be the xy plane, its orientation $\hat{\theta}_t^k$ and its linear velocity $\hat{v}_t^k$, with the subscript t denoting the state value at time step t given all the state observations up to t. When needed the state vector of a particle p of target k is denoted as $\boldsymbol{x}_t^{(k,p)}$.

Each of the particles states of all the targets are updated in this step using the kinematic model shown in equation (1), with the resulting state vector denoted as $\boldsymbol{x}_{t|t-1}^{(k,p)}$.

$$\boldsymbol{x}_{t|t-1}^{(k,p)} = f(\boldsymbol{x}_{t-1}^{(k,p)}) = \begin{cases} x_{t|t-1}^{(k,p)} = x_{t-1}^{(k,p)} + T v_{t-1}^{(k,p)} \cos(\theta_{t-1}^{(k,p)}) \\ y_{t|t-1}^{(k,p)} = y_{t-1}^{(k,p)} + T v_{t-1}^{(k,p)} \sin(\theta_{t-1}^{(k,p)}) \\ \theta_{t|t-1}^{(k,p)} = \theta_{t-1}^{(k,p)} \\ v_{t|t-1}^{(k,p)} = v_{t-1}^{(k,p)} \end{cases} . \tag{1}$$

B. Data Acquisition

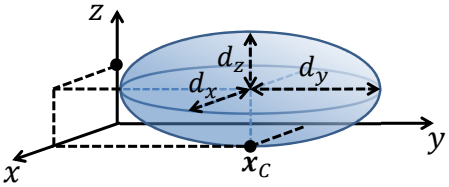
1) Target Visual Model: Instead of using a bidimensional model to define the target shape, the MHPF uses a tridimensional model under the form of an ellipsoid, with center at $\boldsymbol{x}_C = (x_C, y_C, z_C)$, semi-axis lengths in each axis denoted by d_x, d_y, d_z and orientation θ around the zz axis. The semi-axis length d_z and the center z_C coordinate are equal to half the targets height h, which is assumed to be supplied to the algorithm each time a target is detected, together with the remaining axis lengths, and the initial ellipsoid center. Note however that the development of a mechanism by which these quantities are obtained is considered to be out of the scope of this project. Therefore in all of the experiments the target detection information is defined *a priori* by the user.

The vectors $\boldsymbol{x}_C$ and the semi-axis lengths are used, together with θ, to define an ellipsoid [36], through the equation shown in figure 2(a). With the ellipsoid defined it is projected onto

$$(\boldsymbol{x} - \boldsymbol{x}_C^\intercal)^\intercal \, R_z(\theta)^\intercal V R_z(\theta) \, (\boldsymbol{x} - \boldsymbol{x}_C^\intercal) = 1$$

$$V = \begin{bmatrix} d_x^{-2} & 0 & 0 \\ 0 & d_y^{-2} & 0 \\ 0 & 0 & d_z^{-2} \end{bmatrix}$$

(a)



(b)

Fig. 2. (a) Ellipsoid equation (b) Visualization of the ellipsoid in the 3D space

the image plane generating an ellipse [11], [14]. The pixels contained in the resulting ellipse are used to build the target's initial RGB color histogram weighted by an Epanechnikov kernel profile as mentioned in [9]. The number of bins per color channel is defined by the user. This histogram, together with the associated ellipsoid define the target visual model.

2) Image Processing: The idea present in the work of Maggio and Cavallaro [28], of applying the kernel-based tracking method to each particle, is also employed here.

However the number of particles needed is now much larger since the tracking is no longer done directly on the image plane and the state vector size is also larger. Consequently it would be computational burdensome to apply the kernel-based tracking algorithm to all the particles. To handle this problem, the solution adopted was to select a smaller number of particles from each target k by drawing them from the distribution

$$p(\boldsymbol{x}) \sim \sum_{p=1}^{N_p} w_{t-1}^{(k,p)} \delta(\boldsymbol{x} - \boldsymbol{x}_{t|t-1}^{(k,p)}). \tag{2}$$

The resulting particles are iterated, and for each one an ellipsoid is associated with center at $\boldsymbol{x}_C = (x_{t|t-1}^{(k,p)}, y_{t|t-1}^{(k,p)}, h_k)$ and orientation equal to the current state estimate $\hat{\theta}_{t-1}^{k}$. These ellipsoids are then projected into one of the cameras frames and the elliptical region they generate, together with the respective target RGB histogram, are used to apply the KBT method. In order to decide which camera frame is assigned to a sample, information received from the camera scheduling algorithm is used. This tells which cameras have the target k in their FOV according to the current state estimates. One possible strategy would be to choose from these cameras the one that is closest to the target, since it would have less measurement noise as mentioned in section III. However this can produce bad results if, for example, the frame captured by that camera had the target occluded by a scene object(which can be another target) . Therefore using observations from all the cameras viewing a target adds robustness to these kind of situations. Nevertheless closer cameras should be assigned more samples since as aforementioned the measurement noise is lower. To explain how this principle is used to distribute the samples by the valid cameras, suppose that the indices of the cameras having target k in their FOV are the elements of the vector $\boldsymbol{C}_k = (C_k^{(1)}, ..., C_k^{(N)})$. These cameras are ranked according to their distance to target k, with the closest camera receiving a rank r equal to N, which is sequentially decreased by one so that the farthest camera has rank 1. The percentage of samples associated to the frame from camera $C_k^{(i)}$ is set to $\frac{r_i}{N(N+1)/2}$. Has $\frac{r_i}{N(N+1)/2}$ increases with the rank the desired result is achieved. Note that in this process the state of the particles is not taken into consideration for the distribution.

After the application of the KBT for all the samples of a target k, a set of convergence points in the image plane is obtained, which are then back-projected to the WRF yielding an x and y coordinate values. The convergence points whose respective elliptical region produced a RGB histogram with a Bhattacharyya coefficient (BC) below a certain threshold are discarded, which acts as a gating procedure for the data association process that will follow. Finally the remaining obtained coordinates are added to the set $\mathcal{O}_t$ together with those of the targets that have already been iterated. Furthermore, for each point, the RGB histogram obtained in the final iteration of the KBT is also stored. A simple graphical representation of the process described above is shown in figure 3 where the blue squares represent the particles from which the samples are drawn. The different colors intensity intends to depict the several particles weights, and the green curves denote the level curves of the function being maximized during the application of the KBT.

In the case that no elliptical region has a Bhattacharyya coefficient over the threshold an additional procedure is performed. In this case it is considered that the algorithm lost the target, and in order to try to recover the set of smaller particles is redrawn but this time from the distribution

$$p(\boldsymbol{x}) \sim \mathcal{N}\left(\boldsymbol{x} \middle| \frac{\sum_{p=1}^{N_p} w_{t-1}^{k,p} \boldsymbol{x}_{t|t-1}^{(k,p)}}{\sum_{p=1}^{N_p} w_{t-1}^{(k,p)}}, \alpha R^*\right), \tag{3}$$

where R^* is a fixed covariance matrix defined by the user, and α a scaling factor. The process is then repeated with these new particles, while a Bhattacharyya coefficient that surpasses the threshold is not found or the number of allowed retrials is not reached. At each iteration the α value is increased by a predefined amount, therefore increasing the covariance matrix, and consequently the search area. In the extreme case that no point is selected, the maximum Bhattacharyya coefficient obtained in all iterations is computed and the points whose Bhattacharyya coefficient is greater than a percentage of the maximum are added to the set $\mathcal{O}_t$. Note that the only reason why the distribution (3) is used in this procedure, instead of the one in equation (2) is to allow the scaling of the covariance matrix, and consequently the spreading of the particles.

3) Observation Clustering and Data Association: The bidimensional points contained in the set $\mathcal{O}_t$ are expected to be mostly concentrated around the true targets positions. In order to find these positions one can apply a method of density-based clustering [30]. For that a mixture of Gaussians with a number of components equal to the number of targets, N_T, in the current iteration, is used. The Gaussian means $\boldsymbol{\mu}_t^{(n)}, n = 1, ..., N_T$ are then the cluster centers. To estimate the mixture of Gaussians parameters the Expectation-Maximization (EM) algorithm [5] is used.

After the cluster centers are obtained, the goal is to use them as observations for the posterior update of the particle filter. However it is first necessary to associate each observation to a target, which leads to the step of data association. The technique chosen to accomplish this step is based in a Global Nearest Neighbor (GNN) approach [33] where an observation receives a cost according to the target it is assigned to. Then a cost matrix can be built leading to an assignment problem solvable with the Hungarian algorithm. More specifically let $\mathcal{D}$ be a matrix whose entries $\mathcal{D}_{(i,j)}$ are the cost of assigning target i to cluster center j. To define this cost two criteria are considered. First if a cluster center is far away from a target but closest to another, it is more likely that it is associated with the closest one. This assumes that a target, between consecutive iterations is unlikely to drastically change his position.

The second criteria is based on the idea that a cluster center should be more likely associated to a certain target the higher is the BC between that cluster center and the target. However to apply this second criterion, it is necessary to define the BC of a cluster center, relatively to all the other targets. For

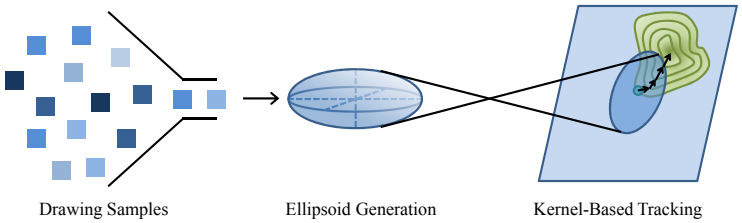


Fig. 3. Graphical representation of the process involved in the acquisition of observations from an image

that consider all the points contained in $\mathcal{O}_t$ which after the execution of the EM algorithm were associated to a certain Gaussian component. Each of these points has an associated RGB color histogram stored previously has mentioned in section IV-B2. The Bhattacharyya coefficients between each of these RGB histograms and the visual histogram models of all the targets are then computed to form the columns of a matrix B. The desired Bhattacharyya coefficients for cluster center $\boldsymbol{\mu}_t^{(n)}$ are then the elements of the vector $\boldsymbol{\rho}_t^{(n)}$ which result from averaging each row of B.

Using the two above criterias the entries of D are then computed as

$$D_{(i,j)} = 1 - (\beta \bar{\mathcal{N}}(\boldsymbol{\mu}_t^{(j)} | \hat{\boldsymbol{x}}_{t-1}^i, \Sigma_d) + \tag{4}$$

$$(1-\beta)\bar{\mathcal{N}}(\boldsymbol{\rho}_t^{(j)} | 1, \sigma_c)), \ 0 \leqslant \beta \leqslant 1, \tag{5}$$

with $\bar{\mathcal{N}}$ denoting an unnormalized Gaussian distribution such that $\bar{\mathcal{N}}(\boldsymbol{\mu}, |\boldsymbol{\mu}, \Sigma_d) = 1$. The parameters Σ_d, β and $\sigma_c, 1-\beta$ are called, respectively, cluster center covariance and cluster distance weight and BC variance and BC weight. The factor β allows to control the weight of each criteria. For example if the targets have a very similar appearance it can be set to a low value since it is not discriminative in that case. On the other hand if the targets have very distinctive colors it can be set to an higher value. Note that the "1−" in expression (5) is needed since the Hungarian algorithm computes the minimum cost assignment and not the maximum.

4) Posterior Update: Up to this point the observations (corresponding to the cluster centers $\boldsymbol{\mu}_t^{(n)}$ described in section IV-B3) consist of target positions. However in order to get measurements of the targets orientation an exhaustive search is employed. This search iterates over all the observations, centering at each one an ellipsoid, whose orientation is changed from $-\pi/2 + \hat{\theta}_{t-1}^k$ to $\pi/2 + \hat{\theta}_{t-1}^k$ at steps of $\pi/36$ radians. For each orientation the resulting ellipsoid is projected onto the image generating an elliptical region whose associated RGB histogram is compared with the one of the respective target. The orientation which gives the higher BC is the one selected to be used as observation.

With the observation vectors complete, the particles weights can now be updated, with the measurement model being defined by the equation $\mathbf{z}_t = [I|\mathbf{0}]\, \boldsymbol{x}_{t|t-1} + \boldsymbol{\nu}_t, \boldsymbol{\nu}_t \sim \mathcal{N}(\mathbf{0}, R)$, with $\mathbf{z}_t$ denoting the observation vector, $\boldsymbol{\nu}_t$ being the measurement noise vector and R the measurement covariance matrix, defined *a priori*. This measurement model allows for one particular kind of proposal distribution for the particle filter. In the simplest particle filter implementations, commonly called the bootstrap particle filter, the importance distribution is chosen to be the prior distribution $q(\boldsymbol{x}_{t|t-1}^{(k,p)} | \boldsymbol{x}_{t-1}^{(k,p)}, \mathbf{z}_{1:t}^{(k)}) = p(\boldsymbol{x}_{t|t-1}^{(k,p)} | \boldsymbol{x}_{t-1}^{(k,p)})$. This simplifies the weights update equation by making $\omega_t^{(k,p)} = Cp(\mathbf{z}_t^{(k)} | \boldsymbol{x}_{t|t-1}^{(k,p)})$. However a common issue with this approach is a phenomenon typically known as degeneracy, that is characterized by the tendency for a few set of particles to have weights much larger than the rest. This leads to a poor approximation of the posterior distribution. One possible solution to overcome this is to choose a better importance sampling. It has been proven [16] that the choice $q(\boldsymbol{x}_{t|t-1}^{(k,p)} | \boldsymbol{x}_{t-1}^{(k,p)}, \mathbf{z}_{1:t}^{(k)}) = p(\boldsymbol{x}_{t|t-1}^{(k,p)} | \boldsymbol{x}_{t-1}^{(k,p)}, \mathbf{z}_{1:t}^{(k)})$ is optimal in the sense of minimizing the variance of the particles weights. In this case the weight update equation also simplifies but to $\omega_t^{(k,p)} = Cp(\mathbf{z}_t^{(k)} | \boldsymbol{x}_{t-1}^{(k,p)})$. The great disadvantage of this choice is that it is usually very difficult to evaluate. However for systems with models of the form

$$\begin{array}{c} \boldsymbol{x}_{t|t-1} = f(\boldsymbol{x}_{t-1}) + \mathbf{w}_t, \ \ \mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, Q) \\ \mathbf{z}_t = H\boldsymbol{x}_{t|t-1} + \boldsymbol{\nu}_t, \ \ \boldsymbol{\nu}_t \sim \mathcal{N}(\mathbf{0}, R) \end{array}, \tag{6}$$

which is the kind of system in this case, it is possible to derive closed form expressions for $p(\mathbf{z}_t | \boldsymbol{x}_{t-1})$ and $p(\boldsymbol{x}_{t|t-1} | \boldsymbol{x}_{t-1}, \mathbf{z}_{1:t})$ as deduced in [16]. This leads to the procedure shown in IV-B4 for the posterior update. The variable $\mathbf{w}_t$ in expression (6) denotes an additive Gaussian noise vector with mean $\mathbf{0}$ and covariance matrix Q and H is the measurement model matrix.

The parameter N_{eff} is the effective number of particles calculated according to

$$\hat{N}_{eff} = \frac{1}{\sum_{p=1}^{N_p} (\omega_t^{(p)})^2}, \tag{7}$$

and N_p is the number of particles. With all the particle weights updated the state estimate for each target is computed using the following estimator:

$$\hat{\boldsymbol{x}}_{t|t}^{(k)} = \frac{\sum_{p=1}^{N_p} \omega_t^{(k,p)} \boldsymbol{x}_{t|t-1}^{(k,p)}}{\sum_{p=1}^{N_p} \omega_t^{(k,p)}}. \tag{8}$$

With the targets state obtained, the multi-target tracker ends and the system proceeds to the next step. At this point the particles state are denoted by $\boldsymbol{x}_t^{(k,p)}$.

V. Experiments

In this experiment the tracking is done in a virtual tridimensional environment with multiple cameras whose calibration

procedure POSTERIOR UPDATE
  for each target k **do**
    for each particle p **do**
      Compute $\mathbf{b}_t^{(k,p)} = H\boldsymbol{x}_{t|t-1}^{(k,p)}$
      Compute $S_t = HQH^\intercal + R$
      Calculate $\Sigma_t^o = Q - QH^\intercal S_t^{-1} HQ$
      Calculate $\mathbf{a}_t^{(k,p)} = \boldsymbol{x}_{t|t-1}^{(k,p)} + \Sigma_t^o H^\intercal R^{-1}(\mathbf{z}_t^k - \mathbf{b}_t^{(k,p)})$
      Update $\omega_t^{(k,p)} = \omega_{t-1}^{(k,p)} \mathcal{N}(\mathbf{z}_t^{(k)} | \mathbf{b}_t^{(k,p)}, S_t)$
    end for
    Normalize $\omega_t^{(k,p)} = \frac{\omega_t^{(k,p)}}{\sum_{p=1}^{N_p} \omega_t^{(k,p)}}$
    for each particle p **do**
      Importance sampling:
      $\boldsymbol{x}_{t|t-1}^{(k,p)} \sim p(\boldsymbol{x}_{t|t-1}^{(k,p)} | \boldsymbol{x}_{t-1}^{(k,p)}, \mathbf{z}_{1:t}^{(k)})$
    end for
    if N_{eff} less than threshold **then**
      Resample particles
    end if
  end for
end procedure

data is known. The main objective is to show the advantages of using an ellipsoid, instead of an ellipse, to model the targets shape. Furthermore the use of observations extracted only from the targets closest camera, in opposition to extract observations from all the cameras observing the target, is also discussed.

A. Setup

In order to perform this experiment a virtual environment was created using Virtual Reality Modeling Language (VRML). An aerial view of this environment, is depicted in figure 4. It represents a parking lot for buses, whose length and width are, respectively, $250[m]$ and $190[m]$. The two entries and two exits are denoted by the black arrows. The red and green arrows and the blue circle, on the other hand, are the xx, yy and zz axis respectively, of the WRF. The pavement of the virtual parking lot is considered to be in the plane $z = 0$. This particular setup has a set of four PTZ cameras

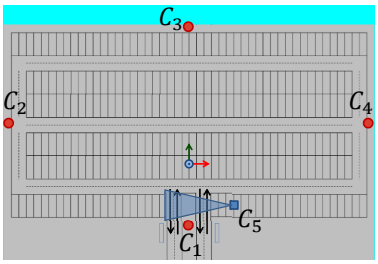


Fig. 4. Aerial view of the virtual environment

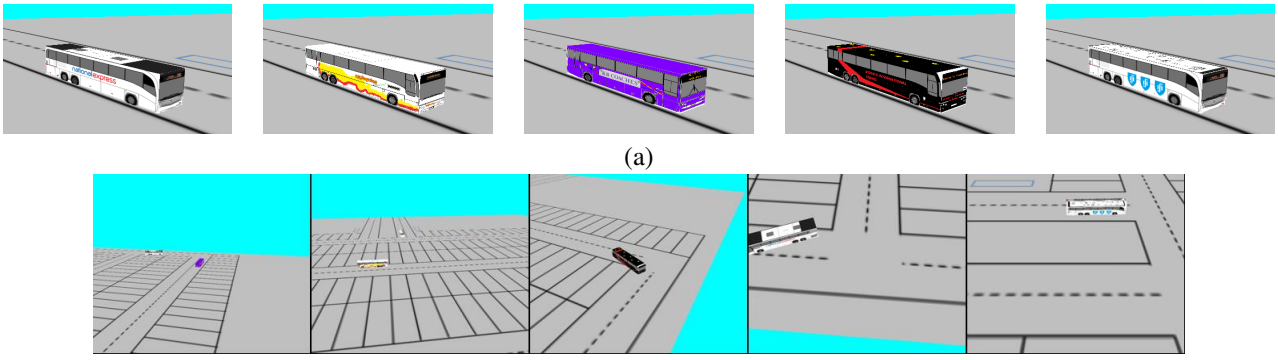
placed, respectively, at positions $(0, -350, 250)$, $(0, 950, 250)$, $(-1300, 300, 250)$ and $(1300, 300, 250)$ which are denoted by the red circles of figure 4. In this same figure, the blue square represents a static camera at location $(250, -250, 200)$ responsible for providing the frames used to build the target visual model when a detection occurs. This static camera is directed towards the two entries as the associated blue triangle intends to show. The resolution of all of these cameras is 704×576. In the case of the PTZ cameras they have a pan and tilt range of $[-\pi, \pi][rad]$ and a maximum pan and tilt velocity of $\pi[rad/s]$. The values chosen are based on the PTZ camera AXIS 215 data sheet, obtained at [3]. In what concerns the surveillance area, described in section III, as no obstacles exist in the parking lot there was no need to define one. The FOV margin, also defined in section III, has a value of 100[pixel] .

The tracking targets chosen are five buses with width, length and height equal to $2[m]$, $10[m]$ and $2[m]$ respectively. Their appearance is shown in figures 5(a) with the buses labeled bus 1 to bus 5 from left to right. The complete simulation spans a $[6min]$ time window in which the parking lot is always opened. During this window the sensors placed at the entrances detect buses 1 to 5 at time steps $t = 8$, $t = 51$, $t = 81$, $t = 92$ and $t = 133$ respectively. In what concerns the buses visual model the ellipsoid dimensions are set equal to the true buses dimensions, and the RGB histogram has 16 bins per color channel.

Regarding the data acquisition process the BC weight was set to 0.4 and consequently the cluster distance weight is 0.6. The cluster center covariance and the BC covariance are respectively $diag(\sigma_c^2, \sigma_c^2)$, $\sigma_c = 20[m]$ and 0.3. The notation $diag(\cdot)$ represents a diagonal matrix. These are the parameters associated to equation (5). For the target recovering procedure a BC threshold of 0.85 and a percentage threshold of 0.85 were chosen. In the KBT algorithm a maximum of 20 iterations are allowed with a convergence threshold of 1[pixel]. The number of auxiliary particles used at this step is 200.

For the multi-target tracker it is considered that a target has stopped if during 2[min] it does not move more than $0.5[m]$ from a position he has at a given iteration. Additionally if the y coordinate of a bus goes below -245 [m] it is considered that it left the parking lot. The number of particles used is 2000, with the resampling threshold set to $2/3$ of the number of particles which is a typical choice [16]. In what concerns the covariance matrix Q of the model (6) it is $diag(\sigma_x^2, \sigma_y^2, \sigma_\theta^2, \sigma_v^2)$, $\sigma_x = 2.5[m]$, $\sigma_y = 2.5[m]$, $\sigma_\theta = \pi/18[rad]$, $\sigma_v = 1.5[m/s]$, and the measurement matrix R is $diag(\rho_x^2, \rho_y^2, \rho_\theta^2)$ with $\rho_x = 0.5[m]$, $\rho_y = 0.5[m]$, $\rho_\theta = \pi/36[rad]$.

Two other methods were also run to compare the impact of the scale adaptation and the difference of extracting observations from one camera or from all the cameras observing a given target. One of them, called "MHPF1", instead of using an ellipsoid to model the target uses an ellipse defined in the image, like in the original KBT [9]. The orientation of this ellipse is also adapted through an exhaustive search. However as this orientation refers to the image plane and not the WRF it is not used as an observation for the particle filter. The ellipse scale remains constant over the entire simulation. In the other method designated "MHPF2" the system only uses observations from the target's closest camera. All the rest stays equal to the MHPF.



(a)

(b)

Fig. 5. (a) Buses 1 to 5, from left to right and (c) View of PTZ cameras C_1 to C_4 and static camera C_5 at time step $t = 133[s]$

B. Experimental Results

In figure 5(b) the frames of all the cameras involved are shown for the time step $t = 133[s]$. Using the labeling of figure 4(a) the frames shown, from the left to the right, are relative to cameras C_1 to C_5. Note that in the case of the frames from the PTZ cameras the buses are not centered in the image as they should be. This happens since the frames were acquired before the cameras were ordered to move.

To measure the trajectory error the mean orthogonal distance is used. The values obtained are shown in table I. The MHPF3 corresponds to the original MHPF and is in the middle of MHPF1 and MHPF2 to allow a more clear comparison. From these results it is possible to notice that the method using the ellipse showed an error much higher than the other two methods almost in all the buses. This result is explained by the fact that with the scale adaptation provided by the ellipsoid, it is less likely that the ellipses associated to two targets overlap. This avoids one of the major problems of the KBT which occurs when two targets are close to each other and have similar visual models. In this case the ellipses may end up following the same target. Even with the clustering process the number of ellipses that converged to the correct buses may not be enough to generate a cluster.

For the results of the MHPF2 and the MHPF the error values are very similar. The MHPF2 however shows better values in all the buses. This is due to the fact that observations from cameras farther from a bus have, generally, more noise associated than those closer to the targets. Therefore they add noise to the process of generating an observation of the target's position. However this increase in the noise is counterbalanced by the fact that using observations of multiple cameras increases the robustness to occlusions, since a target that is occluded in a camera may not be in another.

	Bus 1	Bus 2	Bus 3	Bus 4	Bus 5
MHPF1 ([m])	1.4	7.1	40.7	50.2	10.8
MHPF ([m])	2.8	2.9	0.9	1.2	2.7
MHPF2 ([m])	2.6	1.6	0.7	1.1	2.3

TABLE I
MEAN ORTHOGONAL ERRORS FOR EACH OF THE BUSES USING THE THREE DIFFERENT METHODS

VI. Conclusion and Future Work

In this work is proposed a multi-target visual tracker. The tracker has an hybrid nature in the sense that it combines kernel-based and particle filter trackers. The tracker integrates, in addition, a target tridimensional model. The motivating application for this work is the problem of multi-target tracking using a set of PTZ cameras.

The main contributions of the proposed tracking algorithm are the improvement of the target scale estimation using additional geometric information, and the adaptation to the multi-target scenario of the work of Maggio and Cavallaro [28].

Experimental results show that the proposed tracker is able to handle the challenges without loosing the targets, even in a situation of many similar targets. It is also able to compete with other state of the art algorithms like the MHT obtaining slightly better results.

Beyond the tracker a heuristic camera scheduling algorithm is also presented. For the experimental setup used it proved to be capable of dealing with a situation of more targets than cameras providing satisfactory results.

There are however much space for improvements in the future. In the tracking algorithm, despite its good results, it is not yet applicable to real-time tracking due to the execution time. Furthermore the robustness of the KBT to appearance changes also needs to be improved as it greatly contributes for the trajectory error. It would also be interesting to approach the camera scheduling in the context of the decision theory framework. In particular analyzing the applicability and performance of multi-stage policies like the ones resulting from MAB processes, instead of using myopic strategies of which the heuristic conceived can be considered to be part of.

References

[1] Manya Afonso. Particle filter and extended kalman filter for nonlinear estimation: a comparative study. *IEEE Transactions on Signal Processing*, page 10, 2008.

[2] Sahand Haji Ali Ahmad and Mingyan Liu. Multi-channel opportunistic access: A case of restless bandits with multiple plays. In *Communication, Control, and Computing, 2009. Allerton 2009. 47th Annual Allerton Conference on*, pages 1361–1368. IEEE, 2009.

[3] AXIS. *AXIS 215 PTZ Network Camera Datasheet*, 2014 (accessed June 24, 2014). http://www.axis.com/files/datasheet/ds_215ptz_34462_en_0902_lo.pdf.

[4] Nicola Bellotto and Huosheng Hu. People tracking with a mobile robot: A comparison of kalman and particle filters. In *Proc. of the 13th IASTED Int. Conf. on Robotics and Applications*, pages 388–393, 2007.

[5] Jeff A Bilmes et al. A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. *International Computer Science Institute*, 4(510):126, 1998.

[6] Samuel S Blackman. Multiple hypothesis tracking for multiple target tracking. *Aerospace and Electronic Systems Magazine, IEEE*, 19(1):5–18, 2004.

[7] Tiago Castanheira. Multitasking of smart cameras. *Dissertação para a obtenção do Grau de Mestre em Engenharia Electrotécnica e de Computadores*, 2013.

[8] Zhe Chen. Bayesian filtering: From kalman filters to particle filters, and beyond. *Statistics*, 182(1):1–69, 2003.

[9] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Kernel-based object tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(5):564–577, 2003.

[10] D Davies, P Palmer, and M Mirmehdi? Detection and tracking of very small low contrast objects. 1998.

[11] David Eberly. Perspective projection of an ellipsoid. *Geometric Tools, LLC, http://www. geometrictools, com, Created: Mar*, 2, 1999.

[12] Thomas E Fortmann, Yaakov Bar-Shalom, and Molly Scheffe. Multi-target tracking using joint probabilistic data association. In *Decision and Control including the Symposium on Adaptive Processes, 1980 19th IEEE Conference on*, volume 19, pages 807–812. IEEE, 1980.

[13] J. Gittins, K. Glazebrook, and R. Weber. *Multi-armed Bandit Allocation Indices*. Wiley, 2011.

[14] Nicola Greggio, José António Gaspar, Alexandre Bernardino, and José Santos-Victor. Monocular vs binocular 3d real-time ball tracking from 2d ellipses. In *ICINCO (2)*, pages 67–73, 2011.

[15] Fredrik Gustafsson. Particle filter theory and practice with positioning applications. *Aerospace and Electronic Systems Magazine, IEEE*, 25(7):53–82, 2010.

[16] Anton J Haug. *Bayesian Estimation and Tracking: A Practical Guide*. John Wiley & Sons, 2012.

[17] Alfred O Hero and Douglas Cochran. Sensor management: Past, present, and future. *Sensors Journal, IEEE*, 11(12):3064–3075, 2011.

[18] Alfred Olivier Hero, David Castañón, Doug Cochran, and Keith Kastella. *Foundations and applications of sensor management*. Springer, 2007.

[19] A.K. Hyder, E. Shahbazian, and E. Waltz. *Multisensor Fusion*. NATO science series: Mathematics, physics, and chemistry. Springer Netherlands, 2002.

[20] Simon J Julier and Jeffrey K Uhlmann. A new extension of the kalman filter to nonlinear systems. In *Int. symp. aerospace/defense sensing, simul. and controls*, volume 3, pages 3–2. Orlando, FL, 1997.

[21] RE Kalman. A new approach to linear filtering and prediction problems1. *Journal of basic Engineering*, 82(1):35–45, 1960.

[22] S Kambhampati, K Tangirala, Kamesh R Namuduri, and Sudharman K Jayaweera. Particle filtering for target tracking. In *Wireless Personal and Multimedia Communications*, 2004.

[23] Jungmin Kim, Kyunghoon Jung, Jaeyong Kim, Hajun Song, and Sungshin Kim. Positioning accuracy improvement of laser navigation using unscented kalman filter. In *Intelligent Autonomous Systems 12*, pages 807–816. Springer, 2013.

[24] Y-R Kim, Seung-Ki Sul, and M-H Park. Speed sensorless vector control of induction motor using extended kalman filter. *Industry Applications, IEEE Transactions on*, 30(5):1225–1233, 1994.

[25] Christopher M Kreucher, Alfred O Hero, Keith D Kastella, and Mark R Morelande. An information-based approach to sensor management in large dynamic networks. *Proceedings of the IEEE*, 95(5):978–999, 2007.

[26] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[27] Cody Kwok and Dieter Fox. Map-based multiple model tracking of a moving object. In *RoboCup 2004: Robot Soccer World Cup VIII*, pages 18–33. Springer, 2005.

[28] Emilio Maggio and Andrea Cavallaro. Hybrid particle filter and mean shift tracker with adaptive transition model. In *ICASSP (2)*, pages 221–224, 2005.

[29] Thom Magnusson. State estimation of uav using extended kalman filter. 2013.

[30] Oded Z Maimon and Lior Rokach. *Data mining and knowledge discovery handbook*, volume 1. Springer, 2005.

[31] Luis Montesano, José Gaspar, José Santos-Victor, and Luis Montano. Fusing vision-based bearing measurements and motion to localize pairs of robots. In *International Conference on Robotics and Automation*, 2005.

[32] Jerome Le Ny, Munther Dahleh, and Eric Feron. Multi-uav dynamic routing with partial observations using restless bandit allocation indices. In *American Control Conference, 2008*, pages 4220–4225. IEEE, 2008.

[33] A. Udvarev P. Konstantinova and T. Semerdjiev. A study of a target tracking algorithm using global nearest neighbor approach. In *Proceedings of the International Conference on Computer Systems and Technologies (CompSysTech)*, 2003.

[34] Gregory L Plett. Extended kalman filtering for battery management systems of lipb-based hev battery packs: Part 1. background. *Journal of Power sources*, 134(2):252–261, 2004.

[35] Donald B Reid. An algorithm for tracking multiple targets. *Automatic Control, IEEE Transactions on*, 24(6):843–854, 1979.

[36] Edward J Rzempoluck. *Neural Network Data Analysis Using Simulnet?* Springer, 1998.

[37] Robert A Singer. Estimating optimal tracking filter performance for manned maneuvering targets. *Aerospace and Electronic Systems, IEEE Transactions on*, (4):473–483, 1970.

[38] Chris Snyder, Thomas Bengtsson, Peter Bickel, and Jeff Anderson. Obstacles to high-dimensional particle filtering. *Monthly Weather Review*, 136(12):4629–4640, 2008.

[39] Wiktor Starzyk and Faisal Z Qureshi. Multi-tasking smart cameras for intelligent video surveillance systems. In *Advanced Video and Signal-Based Surveillance (AVSS), 2011 8th IEEE International Conference on*, pages 154–159. IEEE, 2011.

[40] Bjoern Stenger, Paulo RS Mendonça, and Roberto Cipolla. Model-based hand tracking using an unscented kalman filter. In *BMVC*, volume 1, pages 63–72, 2001.

[41] Jaco Vermaak, Simon J Godsill, and Patrick Perez. Monte carlo filtering for multi target tracking and data association. *Aerospace and Electronic Systems, IEEE Transactions on*, 41(1):309–332, 2005.

[42] Long H Vo. Application of kalman filter on modelling interest rates. *Journal of Management*, 1(1):1–15, 2014.

[43] Eric A Wan and Rudolph Van Der Merwe. The unscented kalman filter for nonlinear estimation. In *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*, pages 153–158. IEEE, 2000.

[44] Yibing Wang and Markos Papageorgiou. Real-time freeway traffic state estimation based on extended kalman filter: a general approach. *Transportation Research Part B: Methodological*, 39(2):141–167, 2005.