Multiple Target Tracking With Constrained Motion Using Particle Filtering Methods

A Qualifying Examination Presented in Partial Fulfillment
of the Requirements for the Degree of
Doctor of Philosophy

Ioannis Kyriakides
Department of Electrical Engineering
Arizona State University

Co-Advisors: Dr. Darryl Morrell, Dr. Antonia Papandreou-Suppappola

May 26, 2005

Abstract

Particle filtering has been successfully used in target tracking applications when the state and measurement models are nonlinear and the associated noise is non-Gaussian. Among the most difficult scenarios of target tracking, are multiple target tracking (MTT) problems that require the application of realistic and often multiple kinematic models. Recently algorithms such as the independent partitions (IP) and coupled partitions (CP), that make use of the particle filter, have proven their ability to handle MTT problems effectively, demonstrating the possibilities offered by the particle filter. The particle filter can also use different kinds of additional information that may be available, with the use of likelihood functions and sampling distributions. Such information may arise from targets having constraints in their motion. The ability to incorporate such kinematic behavior into the tracking algorithm can improve tracking performance. The IP and CP assume independence in target motion, therefore ignore information about kinematic constraints, when available. We introduce the constrained motion proposal (COMP) algorithm that uses multitarget proposal densities and motion models that incorporate kinematic constraint information into a particle filter. More specifically, it uses methods of sampling and likelihood functions that take into account motion constraint information. It also introduces variability in partition proposal order, that reduces the effects of erroneously imposed constraints. We compare error performance results of the COMP with those of other proposal methods. We show that the methods used by the COMP make effective use of motion constraint information and, thus, improve tracking performance.

Contents

1 Introduction

Target tracking involves the estimation of the target dynamic state using observations. The solution of this problem might require methods that can deal with highly nonlinear applications such as multiple target tracking (MTT). To tackle this problem, some suboptimal methods have been proposed. The method that this report mainly deals with is the particle filtering method [1], a sequential Monte Carlo technique that has demonstrated its ability of performing successful nonlinear and non-Gaussian tracking, thus, utilizing more accurate models of the dynamics of physical systems and measurement devices. It also enables on-line processing of data, reduces storage requirements and quickly adapts to changing situations. Particle filtering is a Bayesian method that approximates probability densities as point masses (particles). Using Bayes theorem, prior and current knowledge is combined to form an updated version of our knowledge on the state of the targets. This method encompasses all the benefits of Bayesian inference and can be applied using any motion and measurement model.

In recent years, particle filtering has been successful in solving single target tracking problems. This success has been extended to the MTT problem for which new algorithms emerge with increased advantages such as lower computational expense, better performance in difficult scenarios and the ability to utilize measurements from any kind of sensing scheme. In MTT, challenges may include the estimation of the state of multiple moving targets or the number of targets within the observation area at different time steps. MTT needs to utilize measurements arising from the targets' presence in the region of interest or from clutter. However, one must distinguish which measurements to use for the estimation of the state of a certain target and avoid using data from clutter or other targets. Explicitly, this can be achieved using data association methods, which associate data with each target in order to obtain estimates [2–5]. In data association methods, each measurement is associated with a specific target or with clutter. Due to the increasing number of data associations that are possible at each time step, this method can often be computationally prohibitive. Certain methods exist, however, that can alleviate the computational burden such as screening the data and removing data association hypotheses that are less likely. Other methods such as the unified or no data association methods, keep track of the entire joint density of the targets and observations, thus implicitly discriminating among the data for tracking each target by taking into account the useful components of the joint density [6–10]. Methods that do not use data association have been proposed, motivated by the need of simpler algorithms and less computational burden. These methods are based on estimating the entire joint multitarget probability density using a purely Bayesian perspective, avoiding any explicit data association. In [6, 7], the unified data approach is applied using particle filters where each particle includes all the information of interest such as the number of targets present and their states.

In moving target applications, information other than the observed measurements can also contribute to the solution of the target state estimation. Such information may result from the nature of the moving object and its interaction with the environment and other targets. Some examples of such knowledge include the position of obstacles avoided by objects, land avoided by ships, certain paths that restrict motion of objects or the position of other moving objects. Therefore, one can face the challenge of tracking a group of targets moving in a formation or having restrictions in their motion. Such group motion can be the convoy movement, where vehicles should not approach each other closer than a certain distance nor move away too far apart. Another type of constrained motion is when people are moving in a prescribed manner such as the leapfrog motion in military applications. Furthermore, some type of constrained motion can be performed by moving sensors, in order to track targets more effectively.

The purpose of this report is to present current approaches in MTT with no data association. We present existing approaches and introduce new ones, dealing with the MTT problem with kinematic constraints among targets. We propose methods to model group kinematic behavior, that also account for constraints on individual target motion. More specifically, we introduce the Constrained Motion Proposal (COMP) algorithm [11], a general method that can be used to perform tracking effectively considering motion constraints. We investigate the effectiveness the approaches used by the COMP via simulations of constrained motion scenarios.

In this report, we describe the basics of target tracking in Section 2, we examine the particle filtering method for a single target in Section 3 and general MTT approaches in Section 4. We present current algorithms for the MTT problem, that use particle filtering with no data association in Section 5. In Sections 6 and 7 we propose the algorithms for MTT when targets move with constraints. We provide simulation results of the proposed methods and compare them with existing MTT algorithms in Section 8. Finally, we suggest extensions to the COMP and conclude in Section 9.

2 Target Tracking

2.1 Algorithms

Tracking has been used to estimate target information such as position and velocity in many applications [10,12–14,17]. Tracking can be performed in three ways: smoothing, estimation and prediction. In smoothing, an available set of data is used to obtain an estimate of values at a time prior to the latest measurement. In estimation, an estimate is obtained utilizing measurements up to the current time of the estimate. Finally, prediction is used to obtain an estimate at future time steps, based on previous data and any other kind of information available.

The most common approach in target tracking problems is a probabilistic one. In this approach, the kinematics of the target under investigation are given by a motion model that describes the evolution of the state of a target. Using the Bayesian method, a belief on the state of the target given some observation is found using the posterior probability density function (pdf) that is calculated recursively at each time step. An optimal solution to this tracking problem is offered through the Kalman filter [18, 19] that follows the steps of prediction and update. This solution is attainable only when the motion and observation models are assumed known and linear, and when the process and observation noise sequence are assumed Gaussian. The assumptions of linearity and Gaussianity, however, are very restrictive. Thus, in many situations, the Kalman filter fails to provide adequate tracking. Some approximations can be used to modify Kalman filtering for these difficult scenarios [1]. Other suboptimal solutions are the grid-based methods that offer recursion of the posterior, assuming that the state space is discrete. Grid-based methods can become computationally intractable as the state space dimension increases and fail in difficult scenarios and multiple target tracking situations.

Some suboptimal methods exist, however, that relax the strict assumptions of the above methods. The extended Kalman filter (EKF) [20] uses local linearization techniques to describe nonlinear motion and observation models. Since the posterior is approximated by a Gaussian, the Kalman recursion can still be implemented. The unscented Kalman filter (UKF) [21], approximates nonlinearities better by using the unscented transform, which in this case calculates the statistics of Gaussian random variables propagated through nonlinear systems.

Approximate grid-based methods [1] can be useful when the continuous state space can be decomposed into cells. Then, non-Gaussian posterior densities can be approximated using this method. As the state space increases, however, there still exists a prohibitive computational cost. Finally, particle filtering methods [1] require no linearity or Gaussianity assumptions and the computational complexity associated with them is tractable and thus, are becoming increasingly popular.

2.2 Motion and Measurement Models

Kinematic models deal with the evolution of the state of one or many targets. These models are mainly based on the nature of the moving objects. A state vector $\mathbf{x}_k$, usually comprising of position and velocity, describes the state. The state vector evolves as

$$\mathbf{x}_k = \mathbf{f}_k(\mathbf{x}_{k-1}, \mathbf{v}_{k-1}) \tag{1}$$

where $\mathbf{f}_k : \Re^{e_x} \times \Re^{e_\text{v}} \rightarrow \Re^{e_x}$, with e_x and e_v denoting the dimension of $\mathbf{x}$ and $\mathbf{v}$ respectively, is a function of $\mathbf{x}_{k-1}$ and can either be linear or nonlinear. $\mathbf{v}_{k-1}$ is an independent and identically distributed (i.i.d) process noise sequence.

Another component of Bayesian estimation is the measurement model, that characterizes the relationship between the observed data and the state of the targets. The measurement model is represented as

$$\mathbf{y}_k = \mathbf{h}_x(\mathbf{x}_k, \mathbf{n}_k) \tag{2}$$

where $\mathbf{h}_k : \Re^{e_x} \times \Re^{e_n} \rightarrow \Re^{e_y}$ is a known linear or nonlinear function of the state $\mathbf{x}_k$ and the i.i.d measurement sequence $\mathbf{n}_k$.

Examples of measurements include range measurements that provide the distance of the target to the sensor, range rate measurements that measure the rate of change of the distance between the target and sensor, azimuth and elevation angles between the target and sensor in the horizontal and vertical direction, respectively, and sequential images of the position of the target. Some types of sensors often used for target tracking are radar, Infrared, acoustic and optical.

2.3 Bayesian Inference

In order to provide the framework in which the particle filter operates, it is necessary to provide the basics of Bayesian inference. Bayesian inference consists of finding the likelihood function and combining it with the prior information using Bayes theorem to compute a posterior distribution [8]. This will provide a Bayesian recursion useful in many applications.

Let $\mathbf{x}_{1:K} = [\mathbf{x}_1, \ldots, \mathbf{x}_K]$, where the random vectors $\mathbf{x}_k$ with time step $k = 1, \ldots, K$, take values in a state space $\Re^n$. Information about $\mathbf{x}_{1:K}$ is obtained through observations $\mathbf{y}_{1:K} = [\mathbf{y}_1, \ldots, \mathbf{y}_K]$, with each $\mathbf{y}_k$ being within a measurement space $\Re^m$. The likelihood function $p(\mathbf{y}_k|\mathbf{x}_k)$ is obtained using the measurement model as in (2). With the Bayesian perspective, it is possible, in two stages, to recursively obtain the posterior pdf, $p(\mathbf{x}_k|\mathbf{y}_{1:K})$, which represents a belief in the state $\mathbf{x}_k$ at time k based on the measurements up to time k. The two stages are prediction and update.

In the prediction step, it is assumed that the pdf $p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1})$ at time $k-1$ is available. Using the kinematic prior, $p(\mathbf{x}_k|\mathbf{x}_{k-1})$, coming from the motion model given by (1), the posterior $p(\mathbf{x}_k|\mathbf{y}_{1:k-1})$ is obtained, via the Chapman-Kolmogorov equation

$$p(\mathbf{x}_k|\mathbf{y}_{1:k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1})d\mathbf{x}_{k-1}. \tag{3}$$

When a measurement $\mathbf{y}_k$ becomes available at time k, the update stage follows, where the prior is updated using Bayes theorem:

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) = \frac{p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k-1})}{\int p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k-1})d\mathbf{x}_k}. \tag{4}$$

The recursive method described above constitutes the Bayesian inference and shows the ability of this method to accommodate new data in updating information about the state $\mathbf{x}_k$.

Different likelihood functions exist for different kinds of sensors, and the Bayesian inference can be built for many different kinds of motion models.

3 Particle Filtering Method

3.1 Particle Filter Algorithm

Monte Carlo methods are a family of methods that have the common characteristic of solving problems by generating a set of random numbers, based on an educated guess, and calculating the correctness of the guess based on standards given by the problem. Therefore, the Monte Carlo approach is used for problems for which it is not possible to find analytical solutions.

A specific example that proves very useful in tracking problems is the approximation of a pdf by discrete samples. When possible, independent and identically distributed samples are drawn from the target density. Wherever the pdf is peaked, more samples are gathered at that point due to higher probability of occurrence. Therefore, the value of the pdf at discrete points can be estimated by the quantity of samples at that point. The greater the number of samples, the better the density is characterized.

The procedure is, of course, computationally expensive as it requires an adequate amount of discrete samples for representing the density at hand. Through the advancements in computer technology, however, this problem was alleviated.

Sequential Monte Carlo (SMC) methods are used to solve estimation problems by sequentially updating knowledge about the estimates, based on sequentially arriving data. An example of an SMC method is the sequential importance sampling (SIS) [22, 23]; this is also known as particle filtering [24], bootstrap filtering [25], the condensation algorithm [9], interacting particle approximations [26, 27], and survival of the fittest [28].

The main reason for using this SMC technique is to solve problems defined by nonlinear, non-Gaussian scenarios, thus, removing the limitiations imposed by Kalman filtering. The hard to compute posterior density can be approximated by a large set of point estimates that are given weights according to their correctness in describing the density.

The particle filter approximates the posterior density $p(\mathbf{x}_k|\mathbf{y}_k)$ as:

$$p(\mathbf{x}_k|\mathbf{y}_k) \approx \sum_{n=1}^{N} w_k^n \delta(\mathbf{x}_k - \mathbf{x}_k^n) \tag{5}$$

where w_k^n is the weight assigned to particle $\mathbf{x}_k^n$, n is the particle index, $n = 1, \ldots, N$, and N is the number of particles used. The particles $\mathbf{x}_k^n$, are from an appropriate importance density $q(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{y}_k)$ [22, 29]. The optimal importance density has been found to be the one that incorporates information from the kinematics of previous time steps as well as current observations. However, the use of an exact or approximately optimal importance density can be a hard task to perform. If it is not possible to use an importance density that incorporates the current measurements $\mathbf{y}_k$, then the kinematic prior $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ is used as the importance density. The weights of the particles are updated using the weight equation which utilizes the measurement likelihood, the kinematic prior and the importance density

$$w_k^n \propto w_{k-1}^n \frac{p(\mathbf{y}_k|\mathbf{x}_k^n) p(\mathbf{x}_k^n|\mathbf{x}_{k-1}^n)}{q(\mathbf{x}_k^n|\mathbf{x}_{k-1}^n, \mathbf{y}_k)} \tag{6}$$

which reduces to

$$w_k^n \propto w_{k-1}^n p(\mathbf{y}_k|\mathbf{x}_k^n) \tag{7}$$

if the kinematic prior is used as the importance density.

Using the particle filter posterior density approximation in (5) an estimate of the expected value of any function of the state $g(\mathbf{x}_k)$ can be obtained as

$$\mathbf{E}[g(\mathbf{x}_k)] = \int_{\mathbf{x}_k} g(\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_k)d\mathbf{x}_k \approx \sum_{n=1}^{N} w_k^n g(\mathbf{x}_k^n). \tag{8}$$

By increasing the number of particles, the SIS algorithm approaches optimality. By doing so, however, the computational expense of the method increases, especially when dealing with multiple targets. A good choice of importance density usually enables a reduction in the number of particles required.

3.2 The Degeneracy Effect

A problem associated with particle filtering is that the variance of the importance weights increases over time [22]. This implies that only few particles will receive significant weights and thus will not represent the posterior density adequately. Therefore, a large computational effort will be allocated in propagating incorrect samples, with the additional drawback of the few good particles being insufficient to represent the posterior. This is referred to as the particle degeneracy problem [1].

Increasing the number of particles reduces the effect of degeneracy to some extent but has the obvious computational drawbacks. Choosing a good importance density and the use of resampling [1, 30] can reduce the degeneracy effect, as discussed next.

3.3 Choice of Importance Density

The optimal importance density is one that enables us to propose new particles based on both the previous state and the current measurements and is described by $p(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{y}_k)$. This, however, is possible only in particular circumstances: when the state space is a member of a finite set, or when the dynamics are Gaussian [1]. In most cases encountered, however, this is not true, therefore other methods or densities need to be used. A local approximation of the optimal importance density can be obtained with the use of local linearization techniques [22]. A very simple to use importance density is the kinematic prior, $p(\mathbf{x}_k|\mathbf{x}_{k-1})$, which is easy to sample from and its use reduces the weight equation on being dependent on the measurement likelihood. This, however, often causes many particles to be generated that do not represent the posterior adequately, especially when the measurement likelihood is significantly more peaked than the kinematic prior. By the use of the optimal

Table 1: SIR Single Target Tracking Algorithm

- For each particle $n = 1, \ldots, N$
  - Draw particle $\mathbf{x}_k^n \sim q(\mathbf{x}_k | \mathbf{x}_{k-1}^n, \mathbf{y}_k)$
- Compute the weight of each particle as $w_k^n \propto w_{k-1}^n \frac{p(\mathbf{y}_k|\mathbf{x}_k^n) p(\mathbf{x}_k^n|\mathbf{x}_{k-1}^n)}{q(\mathbf{x}_k^n|\mathbf{x}_{k-1}^n, \mathbf{y}_k)}$
- Sample N particles with replacement from the distribution of w_k

importance function or its approximation, the selection of the particles is biased towards the data, therefore achieving lower particle rejection rates and, consequently, the reduction in the number of particles needed to perform successful tracking.

3.4 Resampling

Degeneracy can be solved by the use of resampling, by which the SIS becomes the sequential importance resampling (SIR) particle filter. In this method, the particles that have small weights are eliminated, and the particles with large weights, and that represent the posterior better are propagated to the next step. The particles that survive the resampling procedure are then given equal weights. This procedure can be applied whenever N_{eff} falls below a certain threshold. Different resampling schemes exist. The most preferred method is systematic resampling [31], while other popular schemes are stratified and residual sampling methods [32]. The sampling importance resampling (SIR) single particle filter algorithm using systematic resampling is given in Table 1.

3.5 Single Target Example

To give an illustration on how particle filtering works in practice, we provide a single target tracking trajectory and the attempted estimation of position by the filter. In Figures 1 and 2, the actual track of a moving object in a two-dimensional plane is traced by a continuous line. Each of the groups of particles shown, denoted by points, represent the position estimates of the object at each time step. A weighted average of these position estimates for each group will give the final estimate on position. Figure 1 shows a successful tracking situation. Sometimes, however, the tracker fails to follow the object's actual path, due to clutter, insufficient number of particles, inaccurate motion or observation models etc. In this case, particles are driven away from the actual track. This is illustrated in Figure 2.

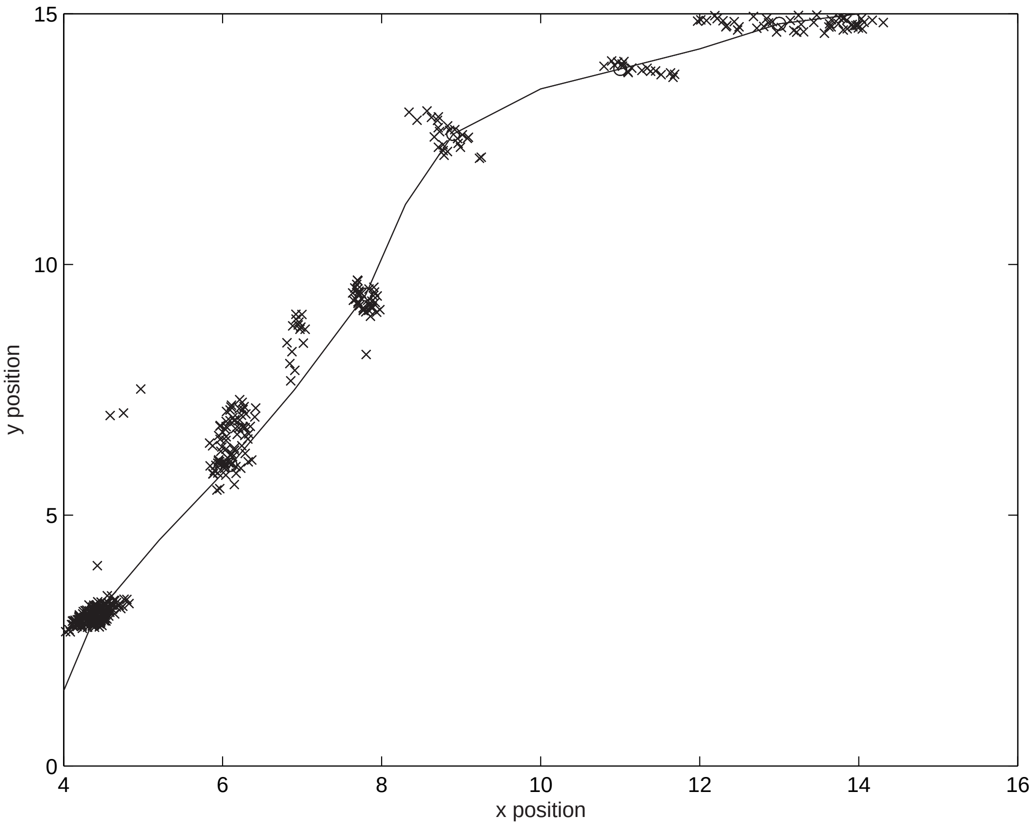

Figure 1: Successful tracking of a target in a two-dimensional plane. The actual track is denoted by a continuous line, while the particle positions at each time step are denoted by points. Each group of points denotes particles at a different time step.

4 Measurements and Multiple Target Tracking

The basic particle filter described in Section 3, deals with the tracking of single targets. In recent years, researchers started to look into the problem of multiple targets, utilizing the benefits of particle filtering with success. The particle filter formulation allows the use of multiple motion models, thus, enabling us to track different multiple targets. Moreover, the ability to utilize different types of measurements, enables us to process data coming from multiple or different types of sensors.

In dealing with multiple targets, one of the main problems one faces is the ambiguity as

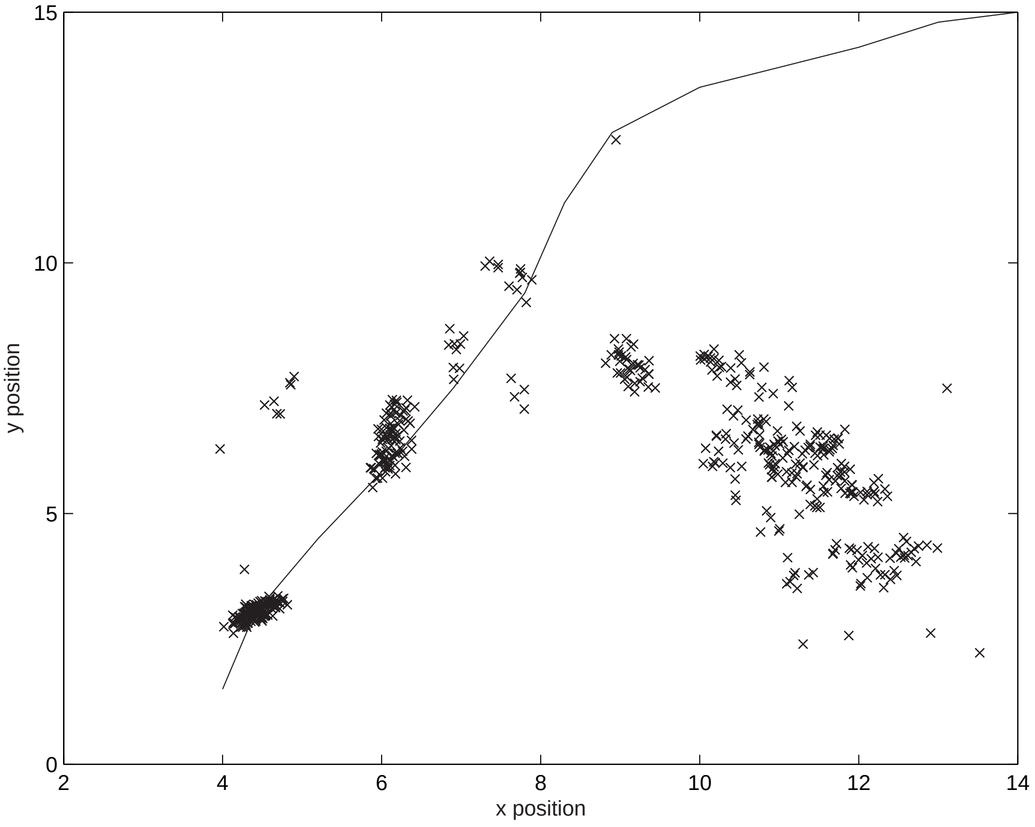

Figure 2: Unsuccessful tracking of a target in a two-dimensional plane. The actual track is denoted by a continuous line, while the particle positions at each time step are denoted by points. Each group of points denotes particles at a different time step.

to the origin of the data received. Measurements may arise from targets or clutter, resulting in false alarms. Furthermore, sensors may miss some valid measurements leading to missed detections. In addition, each measurement that considered valid needs to be associated with the target from which it originated. Another case often involves measurements that arise from two or more targets, or, in a more rare case, one target produces two distinct measurements. Therefore, some methods need to be used in order to deal with clutter and correctly assign observations to targets. Towards this goal, a broad category of data association methods was developed [2, 8]. In data association, measurements are associated with targets, and then these associations are used to produce estimates of the target states. Another technique, the unified data or no data association method, tracks the joint multitarget probability density using a purely Bayesian perspective, avoiding to perform data association explicitly. This aims to remove the complexity associated with data association. Other tasks that need to be performed by multiple target tracking algorithms include accounting for target birth and death which can consist of target splitting or merging, the entry or exit of targets within the measurement space, and confirming of existing tracks. In this section, we provide a review of the data association and the unified approach methods.

4.1 Data Association

At different times, measurements are obtained coming either from targets or false alarms. The measurements are then usually converted to a set of detections, via thresholding. Hypotheses are built by associating each valid measurement with a certain target or labeling it as a false alarm. This describes the general approach of data association techniques. Different approaches that tackle these ambiguous scenarios exist. In Joint Probabilistic Data Association (JPDA) [33–35], each measurement to target association is assigned a probability as to its validity. The state of each target is then estimated taking into account a weighted combination of all the measurements based on their association probability. The computational expense associated with this technique mainly lies in the computation of all the possible associations at each time step. In cases when many of the hypotheses formed are almost equiprobable, this method is not very effective. In a more complicated method, the Multiple Hypothesis Tracking (MHT) method [8, 36–40] the decision on the validity of the hypotheses is delayed until data from future time steps provide a clearer picture of the situation at hand. The actual MHT operation consists of calculating the posterior distribution of the target states given a certain data association and the probability the association is true. The drawback of this technique is that the number of data association hypotheses grows exponentially with the number of measurements over a number of time steps. To combat this situation, two general methods, namely screening and pruning are applied [41]. Screening is applied to restrict the number of measurements that are considered for association for each target. This step is, therefore, applied before constructing the hypotheses. Pruning comes after the construction of the hypotheses to eliminate those that are less likely.

A popular screening technique is gating [36]. In gating, a region is constructed for each target in the measurement space in which the next measurement for that target is expected to be found. There are different sizes and shapes for a gate, according to the situation at hand. Larger regions or gates reduce the probability that a correct measurement is not considered, but potentially allow for more false associations to be established. Smaller gates reduce association complexity, but are prone to missassociations.

In clustering screening method [41], targets are grouped according to the measurements that they may be associated with. For example, two groups of targets that are close spatially face a strong data association problem. The data, however, pertaining to each group can be considered for data association only within that group. Thus, the data association issue is partitioned, mitigating challenges in assigning measurements to targets. A form of gating can be used in this case to facilitate the decision on the clustering of targets.

A third screening method is classification. In [41], four categories of targets are defined that are classified according to the confidence level of the correctness of the track assigned to them. A threshold is set on the amount of that confidence. Therefore, going from a higher confidence level to a lower one, targets can be grouped as confirmed, intermediate, tentative and born.

In pruning, the basic idea is to set a probability threshold and allow only the associations for which the probability exceeds that threshold to be carried to the next time step. With this method, the associations are not allowed to increase exponentially with time; this makes the MHT computation tractable.

A non-probabilistic method is the Nearest Neighbor (NN) [42] approach. In this method, a measurement is predicted for each target, and the target is associated with the measurement that is spatially or otherwise closer to its predicted measurement than all other measurements. This is a simple method to use. Its non-probabilistic approach on making association decisions, however, causes it to fail to track adequately in dense clutter environments.

Some algorithms for MMT using particle filtering are found in the literature. In [2], the probabilistic multiple hypothesis tracker (PMHT) employs particle filtering to perform MHT. In [43, 44], the JPDA, which uses a weighted sum over all association probabilities, is combined with particle filtering.

4.2 Unified Approach or No Data Association

Data association approaches, in addition to being computationally demanding, are restrictive as to the sensor models and types of data that can be used. Some types of measurements need to be thresholded so that they contain valid measurements and false alarms and, in most cases, there is an assumption that each detection originates from only one target.

Many sensor schemes, however, provide measurements for which data association is not possible. An example of this is measurements from sensors that detect energy levels. If it is assumed that each target produces an energy level that decays with distance, then the energy level received by each sensor would be the sum of the energy levels, emanating from all the targets, at the point in space where the sensor exists. To resolve this issue, a unified tracking method can be used [8]. With this method, it is possible to calculate the joint multitarget probability density [45] using the measurement state as a whole instead of having to threshold it in order to assign each measurement to a target. In a data association method thresholding would have to be used together with other assumptions that would degrade the performance of the tracker in a possibly large extent. With the unified model such measurements can be used as is. Thus, one utilizes the maximum information received from the tracking environment using no data association.

In [6, 10, 46], some approaches with no explicit data association are described. The independent partition (IP) particle filter [6] partitions the state space to obtain a number of partitions equal to the number of targets, and uses the particle filtering method to propose each partition separately. The proposed partitions are combined to propose particles. The particles are then weighted according to the whole measurement space; thus, data association is not explicitly performed.

The IP algorithm is much simpler and less computationally expensive than the data association approaches. This algorithm, however, assumes independence between the targets. When the targets are spatially or otherwise close to each other, the independence assumption is not valid. In [7, 47], the coupled partitions (CP) algorithm is provided, in which a number of realizations are proposed for each partition, and one of these realizations is selected for each partition of each particle. The CP algorithm has the advantage of performing well when the independence assumption between targets does not hold. However, it is very computationally expensive. An adaptive method, the adaptive partition (AP) method, is also proposed in [7, 47], where switching between using the IP and CP is performed based on whether the investigated targets are assumed independent or not, at various time instances. This alleviates the computational burden of using the CP alone.

5 Particle Filter Algorithms for Multiple Target Tracking

Before describing the IP and CP algorithms in detail, we first provide the dynamic model for MTT.

5.1 State Space Formulation

We consider the motion of a fixed number of targets moving in a two dimensional plane. The dynamics of the targets are modeled using the same motion model, assuming nearly constant velocity. The targets are simultaneously tracked by three sets of passive sensors providing angle-only measurements.

5.1.1 Motion Model

The system model describes the dynamics of L targets moving on a two dimensional plane. The motion of the targets is considered to be linear and independent from other targets. The state vector for the lth target, $l = 1, \ldots, L$, comprises of four components:

$$\mathbf{x}_{l,k} = \begin{bmatrix} x_{l,k} \\ \dot{x}_{l,k} \\ y_{l,k} \\ \dot{y}_{l,k} \end{bmatrix}$$

corresponding to the position in the x and y directions, $x_{l,k}$, $y_{l,k}$ and the velocity in the x and y directions, $\dot{x}_{l,k}$, $\dot{y}_{l,k}$. The subscript k, $k = 1, \ldots, K$, denotes the time step, and K is the time of the completion of the tracking procedure. The motion is formulated as:

$$\mathbf{x}_{l,k} = \mathbf{F}\mathbf{x}_{l,k-1} + \mathbf{Q}^{1/2}\mathbf{v}_{k-1} \tag{9}$$

where

$$\mathbf{F} = \begin{bmatrix} 1 & \tau & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \tau \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

and τ is the time difference between state transitions. The matrix $\mathbf{Q}$ is the process noise covariance matrix that is used to model the acceleration in the x and y directions and $\mathbf{v}_k$ denotes a zero mean, unit variance Gaussian process.

The state vector describing the dynamics of all targets is a concatenation of the individual state vectors of each target

$$\mathbf{X}_k = [\mathbf{x}_{1,k}^T \ \mathbf{x}_{2,k}^T \ \ldots \ \mathbf{x}_{L,k}^T]^T \tag{10}$$

where $\mathbf{x}^T$ denotes the transpose of $\mathbf{x}$. We will refer to each component $\mathbf{x}_{l,k}$ for $l = 1, \ldots, L$ of the state vector $\mathbf{X}_k$, as a partition.

5.1.2 Measurement Model

S sensors, positioned in the same plane as the targets, obtain angle measurements indicating the relative angle between each sensor and the corresponding target. Each sensor observes I cells as shown in Figure 3. Each cell covers a certain angle in the measurement space, which is equal for all cells, and yields a measurement $y_{s,i,k}$, $s = 1, \ldots, S$, $i = 1, \ldots, I$. Here, s corresponds to the sensor number, and i to the cell number. This measurement takes values of 1 or 0 according to whether a target is in a cell. The resulting output of each sensor at each time step k is denoted as $\mathbf{y}_{s,k} = [y_{s,1,k} \; \ldots \; y_{s,I,k}]^T$, and the total output of all sensors as $\mathbf{Y}_k = [\mathbf{y}_{1,k}^T \; \ldots \; \mathbf{y}_{S,k}^T]^T$.

Next, we derive the likelihood $p(\mathbf{Y}_k|\mathbf{X}_k)$ of the measurements $\mathbf{Y}_k$ conditional to the true target state $\mathbf{X}_k$. Each cell receives energy that originates from targets that are within its angle of observation and noise. The decision on the measurement value that is output by each cell is obtained based on a certain probability of false alarm (P_{FA}). Assuming that the energy measurements are Rayleigh-distributed with a signal-to-noise ratio (SNR) denoted by λ that is equal for all targets, the probability of detection (P_D) in a given cell i is, [7]

$$P_D(n_i) = P_{FA}^{\frac{1}{1+n_i\lambda}} \tag{11}$$

where n_i is the number of targets in cell i.

The likelihood related to each cell at time k is defined by

$$p(y_{s,i,k} = 1|\mathbf{X}_k) = \begin{cases} P_{FA} \text{ if } \mathbf{X}_k \text{ has no targets in cell } i \\ P_D(n_i) \text{ if } \mathbf{X}_k \text{ has } n_i \text{ targets in cell } i \end{cases}$$

Each cell is assumed to obtain measurements independently. Therefore, the likelihood relative to sensor s is obtained by the product of the likelihoods for each cell:

$$p(\mathbf{y}_{s,k}|\mathbf{X}_k) = \prod_{i=1}^{I} p(y_{s,i,k}|\mathbf{X}_k). \tag{12}$$

Finally, the total likelihood is given by the product of the likelihoods of all the sensors:

$$p(\mathbf{Y}_k|\mathbf{X}_k) = \prod_{s=1}^{S} p(\mathbf{y}_{s,k}|\mathbf{X}_k). \tag{13}$$

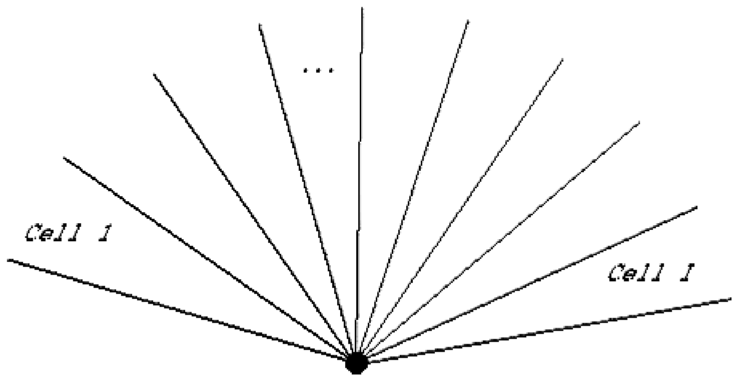

Figure 3: Schematic of a typical sensor used. Each cell covers a certain angle, reporting a detection or no detection at its output.

5.2 Particle Filtering Algorithms

5.2.1 IP Algorithm

The SIR single target particle filter [1] extends directly to the SIR multitarget particle filter [7], using the definitions of the motion and measurement models in Section 5.1. This filter proposes particles that consist of partitions, each representing one of the targets of interest. The partitions in a particle, however, are proposed simultaneously and placed into particles without any additional assessment of their validity. Therefore, many of the particles that are built receive low weights at the particle weighing step. For example, some of the partitions of the particle might be very good estimates of the state vector of certain targets, while other partitions in the same particle can be bad estimates of other target's state vectors. Thus, many particles can be composed of both good and bad estimates. This results in particles receiving an overall low weight, due to the bad partitions.

The IP algorithm [6], offers a method of constructing better particles. This is achieved by independently proposing partitions for each particle, weighing them and choosing the partitions that receive the best weights by sampling from the weight density of each partition. This constitutes the proposal subroutine. More specifically, the partitions are proposed via a single partition importance density of our choice and weighted by a function that contains useful information coming from the measurements. The chosen partitions are then placed into particles, and the resulting particles are weighted. As a further advantage, the overall better particles have greater chances of being propagated to the next time step, if resampling is used. This results in a great reduction in the number of particles required for representing the posterior adequately and an improvement in overall performance when compared to the SIR multitarget particle filter.

Usually an approximation of a density that involves single target measurement information is used as a weighing function in the proposal subroutine. A simple choice is an approximation to the single target measurement likelihood function. Note that each partition can be weighed individually because all the measurements are well separated in the sensor space. Then each partition will receive weights which are due only to the part of the joint posterior that represents the target corresponding to that partition. This is true even if, in practice, it is weighted against the whole posterior density. This is important as partitions can be proposed performing no data association.

Next, the assumptions that are used to implement the IP algorithm are presented more explicitly. The first assumption is that the dynamics of the particles are independent. Thus, the state transition density of the combination of the partitions, each representing the state of a targets, can be factorized as:

$$p(\mathbf{X}_k^n|\mathbf{X}_{k-1}^n) = \prod_{l=1}^{L} p(\mathbf{x}_{l,k}^n|\mathbf{x}_{l,k-1}^n). \tag{14}$$

If each of these factors are used as the single proposal density, it is possible to propose each partition of each particle, corresponding to a certain target, using the prior density of that target. Also, the joint posterior distribution can be factorized as

$$p(\mathbf{X}_k|\mathbf{Y}_k) = \prod_{l=1}^{L} p(\mathbf{x}_{l,k}|\mathbf{Y}_k). \tag{15}$$

The above proceeds from the assumption that the targets are not coupled in the sensor space,

Since the joint posterior distribution of the multitarget state can be factorized into independent components, each corresponding to one of the partitions, then, for each particle, the partitions can be constructed independently. Therefore, partitions are proposed independently through the kinematic prior, $p(\mathbf{x}_{l,k}^n|\mathbf{x}_{l,k-1}^n)$, $l = 1, \ldots, L$, and eventually weighted. Therefore, the weight equation for each partition $l = 1, \ldots, L$ reduces to

$$w_{l,k}^n \propto p(\mathbf{Y}_k|\mathbf{x}_{l,k}^n). \tag{16}$$

After resampling, that involves obtaining a sample j of the partition l from the distribution

Proposed Particles		Proposed particles after partition sampling
Particle 1 : $\{\mathbf{x}_1^1, \mathbf{x}_2^1, \mathbf{x}_3^1\}$		Particle 1 : $\{\mathbf{x}_1^3, \mathbf{x}_2^1, \mathbf{x}_3^2\}$
Particle 2 : $\{\mathbf{x}_1^2, \mathbf{x}_2^2, \mathbf{x}_3^2\}$	$\longrightarrow$	Particle 2 : $\{\mathbf{x}_1^1, \mathbf{x}_2^3, \mathbf{x}_3^2\}$
Particle 3 : $\{\mathbf{x}_1^3, \mathbf{x}_2^3, \mathbf{x}_3^3\}$		Particle 3 : $\{\mathbf{x}_1^1, \mathbf{x}_2^2, \mathbf{x}_3^3\}$

Table 2: Schematic of three particles, $n = 1, \ldots, 3$, with three partitions, $l = 1, \ldots 3$: $\mathbf{x}_{1,k}^n$, $\mathbf{x}_{2,k}^n$ and $\mathbf{x}_{3,k}^n$, where the time subscript k was omitted above. The particles are formed after independent proposal of individual partitions and sampling from the weight distribution of partitions. Here, the partitions shown on the left received good weights and were more likely to be chosen, thus forming particles that overall represent the posterior well.

of $w_{l,k}$, the weight $w_{l,k}^j$ received by each partition $l = 1, \ldots, L$ becomes the bias $b_{l,k}^n$ for that partition, corresponding to particle n. Therefore, $b_{l,k}^n{=}w_{l,k}^j$.

With the IP, one may permute the order of partitions corresponding to the same targets between particles, constructing particles with good partitions. This is demonstrated in Table 2.

Another benefit of the IP technique is that the choice of the particles is biased towards the measurements. Since this is a biased sampling scheme, the bias of the partitions are retained and used in the final weighing step. To express the procedure in a mathematical form, a derivation of the weighing equation for the particles is presented. The weight of the particle $n = 1, \ldots, N$ will be:

$$w_k^n \propto w_{k-1}^n \frac{p(\mathbf{Y}_k|\mathbf{X}_k^n)p(\mathbf{X}_k^n|\mathbf{X}_{k-1}^n)}{q(\mathbf{X}_k^n|\mathbf{X}_{k-1}^n, \mathbf{Y}_k)} \tag{17}$$

where $q(\mathbf{X}_k^n|\mathbf{X}_{k-1}^n, \mathbf{Y}_k)$ is the proposal density used to propose each of the particles. Recall that each of the partitions of the particle was proposed by the kinematic prior of the target it represents and weighted with the whole posterior. This weight biased its selection for placement in the particle. Therefore,

$$q(\mathbf{X}_k^n|\mathbf{X}_{k-1}^n, \mathbf{Y}_k) = \prod_{l=1}^{L} q(\mathbf{x}_{l,k}^n|\mathbf{x}_{l,k-1}^n, \mathbf{Y}_k) \tag{18}$$

and

Table 3: Independent Partitions Algorithm [6]

```
• For each partition $l = 1, \dots, L$
    – For each particle $n = 1, \dots, N$
        * Sample $\mathbf{x}^n_{l,k} \sim p(\mathbf{x}_{l,k}|\mathbf{x}^n_{l,k-1})$
        * Compute $w^n_{l,k} = w^n_{l,k-1} p(\mathbf{Y}_k|\mathbf{x}^n_{l,k})$
    – Normalize $w_{l,k}$
    – Resample by choosing a partition $j$ from the distribution of $w_{l,k}$
      with replacement
    – Keep the bias of each sample as $b^n_{l,k} = w^j_{l,k}$
• For each particle $n = 1, \dots, N$
    – Compute $w^n_k = w^n_{k-1} \frac{p(\mathbf{Y}_k|\mathbf{X}^n_k)}{\prod_{l=1}^L b^n_{l,k}}$
```

$$q(\mathbf{x}^n_{l,k}|\mathbf{x}^n_{l,k-1}, \mathbf{Y}_k) = b^n_{l,k} p(\mathbf{x}^n_{l,k}|\mathbf{x}^n_{l,k-1}). \tag{19}$$

Combining (62) and (63), the weight of each particle becomes

$$w^n_k \propto w^n_{k-1} \frac{p(\mathbf{Y}_k|\mathbf{X}^n_k) p(\mathbf{X}^n_k|\mathbf{X}^n_{k-1})}{(\prod_{l=1}^L b^n_{l,k})(\prod_{l=1}^L p(\mathbf{x}^n_{l,k}|\mathbf{x}^n_{l,k-1}))} \tag{20}$$

which, using (14), becomes

$$w^n_k \propto w^n_{k-1} \frac{p(\mathbf{Y}_k|\mathbf{X}^n_k)}{\prod_{l=1}^L b^n_{l,k}}. \tag{21}$$

The steps of the IP method are summarized in Table 3.

5.2.2 CP Algorithm

The assumptions that are used for the IP are true only when the targets are sufficiently separated in the sensor space. This provides us with an uncorrelated measurement likelihood. When, however, some targets are coupled or close in sensor space, then their partitions cannot be rearranged in the way explained above. One way to solve this is to rearrange only the uncoupled partitions, and treat the coupled and ones as a group, transfer them, if necessary, to another particle without separating them. The drawback of this is that if many targets are coupled for most of the time this would cause the performance of the IP algorithm to degrade to the performance of the SIR multitarget filter. Another method that does not require the assumptions used by the IP scheme is the CP method [7].

As the name suggests, the CP algorithm was designed to handle partitions that are coupled. The main objective is to built good particles using the best possible partitions. The best possible partitions are those that receive good weights, when weighted with the joint posterior. The CP algorithm differs from the IP in that it proposes each partition of each particle M times, and out of those M outcomes it chooses the best outcome to be placed in the particle. Thus, the particles do not interact by exchanging their partitions but rather each particle is built by choosing from a pool of candidates for each of its partitions and remains intact for weighing. This feature is illustrated in Table 4.

This feature enables the CP algorithm to handle coupled targets, since the independence assumption of the IP is no longer necessary. The proposal of each partition M times, however, makes the CP computationally burdensome, especially with an increase in the number of interacting targets. An adaptive proposal (AP) method suggested in [7] switches between using the IP and the CP in proposing partitions, depending on whether the targets are coupled or not. This results in a reduction in computational complexity that is proportional to the time that the targets are well-separated and the number of uncoupled targets. The equation for weighing the particles in the CP method is constructed exactly in the same way as in the case of the IP. The details of the CP algorithm are given in Table 5.

Another concern regarding the structure of the particles is the ordering of their partitions. The proposal of each partition and the weighing of the whole particle are invariant to permutations of partitions inside the particle. Therefore, the outcome of the CP algorithm are particles that do not necessarily have their partitions identically placed according to target number. Partitions that are possibly swapped within the particle correspond to targets that were coupled at some point. This is due to the fact that when targets become coupled, some of their properties may become very similar. Therefore, differently labelled partitions may represent the wrong targets with no penalty from the weight equation, since the quantities they represent are almost the same. This event is referred partition swapping [7]. This would cause errors in the final multitarget estimate, the calculation of which relies on the fact that partitions are identically placed according to the target number for all particles. To correct partition swapping, the K-means algorithm is used before estimation to group

$$\begin{tikzcd}
\mathbf{x}_1^1(1) & \mathbf{x}_2^1(1) & \mathbf{x}_3^1(1) \\
\mathbf{x}_1^1(2) & \mathbf{x}_2^1(2) & \mathbf{x}_3^1(2) \\
\mathbf{x}_1^1(3) & \mathbf{x}_2^1(3) & \mathbf{x}_3^1(3) \\
\mathbf{x}_1^1(4) & \mathbf{x}_2^1(4) & \mathbf{x}_3^1(4) \\
\mathbf{x}_1^1(5) \arrow[d] & \mathbf{x}_2^1(5) \arrow[d] & \mathbf{x}_3^1(5) \arrow[d] \\
\mathbf{x}_1^1(3) & \mathbf{x}_2^1(4) \arrow[d] & \mathbf{x}_3^1(5) \\
 & \{\mathbf{x}_1^1(3), \mathbf{x}_2^1(4), \mathbf{x}_3^1(5)\} &
\end{tikzcd}$$

Table 4: For the choice of the partitions composing each particle, each partition is chosen from a pool of candidates proposed for that partition. A single candidate, with the highest weight is chosen for each partition.

all partitions corresponding to each target by placing them in the same order inside each particle. An example of partition swapping and the subsequent correction by the K-means algorithm are shown in Figure 4 and Figure 5, respectively. For the details concerning the K-means refer to [7].

6 Tracking Targets Moving With Kinematic Constraints

In a multitarget scenario, one can face the challenge of tracking a group of targets moving in a formation or having restrictions in their motion. This motion can be described by equations or simply by some rules. An example of constrained motion is convoy movement [15], where vehicles should not approach each other closer than a certain distance nor move away too far apart. Another example is people moving in a prescribed manner such as the leapfrog motion in military applications [16]. Furthermore, some type of constrained

Table 5: Coupled Partitions Algorithm [7]

```
• For each partition $l = 1, \ldots, L$
    – For each particle $n = 1, \ldots, N$
        ∗ For each proposal $m = 1, \ldots, M$
            · Sample $\mathbf{x}_{l,k}^{n}(m) \sim p(\mathbf{x}_{l,k}|\mathbf{x}_{l,k-1}^{n})$
            · Compute $w_{l,k}^{n}(m) = p(\mathbf{Y}_k|\mathbf{x}_{l,k}^{n}(m))$
        ∗ Normalize $w_{l,k}^{n}$ and sample from it once
        ∗ Normalize $w_{l,k}$
        ∗ Resample by choosing a partition $j$ from the distribution of $w_{l,k}$ with replacement
        ∗ Keep the bias of the sample as $b_{l,k}^{n} = w_{l,k}^{j}$
• For each particle $n = 1, \ldots, N$
    – Compute $w_k^n = w_{k-1}^n \frac{p(\mathbf{Y}_k|\mathbf{X}_k^n)}{\prod_{l=1}^{L} b_{l,k}^n}$
```

motion can be performed by moving sensors in order to track targets more effectively.

Knowledge of such a kinematic behavior can be very useful as it can improve the tracking performance if incorporated properly in the kinematic model and proposal distribution. In fact, in certain scenarios, not taking into account such target motion behavior could result in lost tracks. The IP and CP methods assume independent kinematics among targets, therefore, they omit such useful information. Some books and papers have dealt with the subject of group target motion [17,48] a problem similar to tracking points on an extended object [49–51]. These approaches model target motion as having independent components that are superimposed by a group effect. However, such methods deal with the problem of targets moving together as a group, and not the more general case of constraint motion where targets may move in their own paths but still influence each other's motion.

Here, we investigate alternative methods to model group kinematic behavior that also accounts for constraints on individual target motion. We propose the constrained motion proposal (COMP) algorithm, a general method that can be used to perform tracking effectively considering motion constraints. Two variations of the method are presented for the

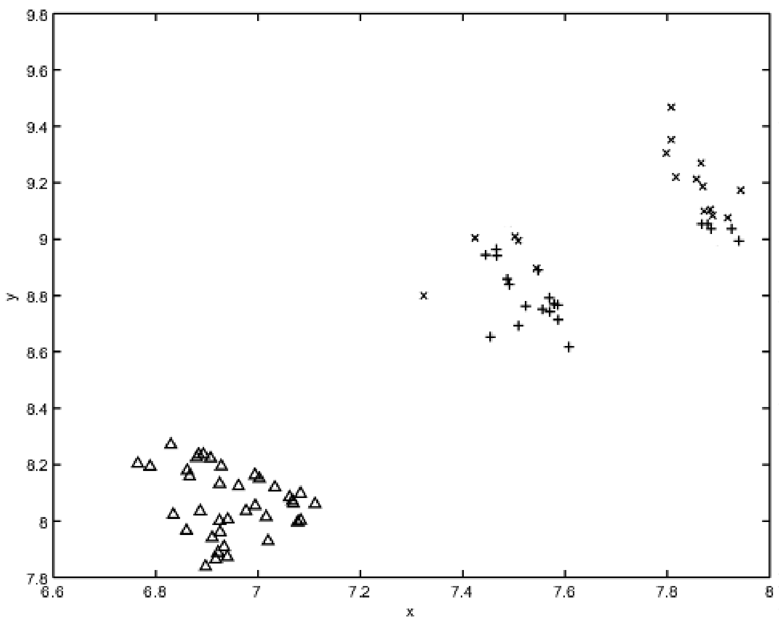

Figure 4: A situation of partition swapping. Each particle is built by 3 partitions tracking 3 targets. The symbols $\triangle$, $+$ and $\times$ correspond to each of the targets. In this situation, some $+$ and some $\times$ partitions are mislabeled inside their particles because of the coupling of the targets they represent. The K-means needs to be applied to reorder them accordingly.

case when the motion of the targets follows some rules and can be easily extended to the case when the motion can be described by equations.

6.1 The COMP Algorithm

The COMP algorithm [11], similarly to the IP, assumes that the joint multitarget posterior density factorizes into the measurement likelihood and the individual prior densities of the targets as described in [6, 7]. However, the COMP takes into account the dependency of the motion of the targets on constraints by introducing importance densities incorporating the constraints and the constraint likelihood function that reflects the constraints on the joint density. The idea of using the constraint likelihood function is derived from the use of the land avoidance function described by [8]. The proposal density tries to approximate the joint posterior. Thus, it considers both the measurements and the constrained kinematics in proposing particles.

The measurements are incorporated into the proposal as with the IP. The kinematics, however, need to include the constraints that each target has in its motion. This is done

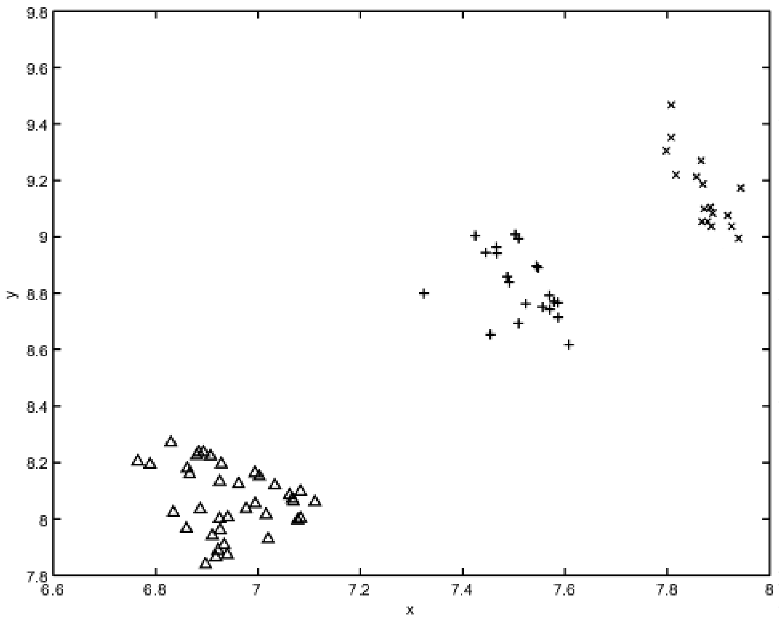

Figure 5: The K-means algorithm has corrected the situation of partition swapping in Figure 4 by labeling the partitions $\times$ and $+$ in the correct way.

using two similar approaches. The first approach, that we call Reproposed Gaussian COMP (RCOMP), proposes partitions repeatedly until the desired constraints are satisfied (see Table 6). The second approach, that we call Truncated Gaussian COMP (TCOMP), proposes partitions by truncated Gaussian distributions, where the truncation points correspond to bounds in target motion (see Table 7). For both approaches, the constraints imposed on partitions of certain targets depend on the nature of a target and its environment as well as on other targets with which they interact. We will refer to the former as self-constraints and the latter as group constraints.

Using either of these two approaches results in proposals that are constructed sequentially for each partition and often depend on the values of the state vectors of previously proposed partitions. These values, however, could be erroneous. Therefore, variability in the order of proposing the partitions for each particle is introduced. For example, proposed partition samples representing one target may take values which are far from the target's true value, therefore providing incorrect constraints for the subsequently proposed partitions. As a consequence, other partitions will diverge from the true value of their respective states due to one incorrectly tracked target. The effect of this may be easily reduced by

Table 6: Reproposed Constrained Motion Proposal Algorithm

```
• For each permutation $u = 1, \dots, L!$
    – For each index $r = 1, \dots, L$
        ∗ The current partition is $\ell_{r,u}$
        ∗ For each particle $n = \frac{(u-1)N}{L!} + 1, \dots, \frac{uN}{L!}$
            · Obtain $\check{\mathbf{C}}^n_{\ell_{r,u},k}$ by choosing the strictest bounds available for the partition
            · Sample $\mathbf{x}^n_{\ell_{r,u},k} \sim q(\mathbf{x}_{\ell_{r,u},k}|\mathbf{x}^n_{\ell_{r,u},k-1}, \check{\mathbf{C}}^n_{\ell_{r,u},k}, \{\mathbf{x}^n_{\ell_{r',u},k},\ \ell_{r',u} : r' < r\})$
            · If constraints not met, return to previous step
            · Compute $w^n_{\ell_{r,u},k} \propto p(\mathbf{Y}_k|\mathbf{x}^n_{\ell_{r,u},k})\mathbf{I}(\mathbf{x}^n_{\ell_{r,u},k}, \check{\mathbf{C}}^n_{\ell_{r,u},k})$
        ∗ Normalize $w_{\ell_{r,u},k}$
        ∗ For each particle $n = \frac{(u-1)N}{L!} + 1, \dots, \frac{uN}{L!}$
            · Obtain a sample $j$ from the distribution of $w_{l,k}$ with replacement
            · Retain $\mathbf{x}^n_{\ell_{r,u},k} = \mathbf{x}^j_{\ell_{r,u},k}$ and $b^n_{\ell_{r,u},k} = w^j_{\ell_{r,u},k}$
            · Retain $\{\mathbf{x}^n_{\ell_{r',u},k} = \mathbf{x}^j_{\ell_{r',u},k} : r' < r\}$ and $\{b^n_{\ell_{r',u},k} = w^j_{\ell_{r',u},k} : r' < r\}$
• For each particle $n = 1, \dots, N$
    – Compute $w^n_k \propto w^n_{k-1} \frac{p(\mathbf{Y}_k|\mathbf{X}^n_k)\mathbf{I}(\mathbf{X}^n_k, \mathbf{C}^n_k)}{\prod_{l=1}^{L} b^n_{l,k}}$.
```

changing the ordering in which we propose partitions among subsets of particles.

In Section 6.2, we present the kinematic density and measurement model used in the algorithms. In Section 6.3, we provide details concerning the way the variability in the order of proposal is achieved and the formation of the region of constraints. In Section 6.4, we derive the multitarget importance density utilized by the RCOMP and the TCOMP and describe the RCOMP and TCOMP algorithms in detail. In Section 6.5, we derive the partition and particle weight equations. In Section 6.6, we provide a simple example showing how particles are built using the COMP. Finally, in Section 6.7 we provide a detailed algorithm description.

Table 7: Truncated Gaussian Constrained Motion Proposal Algorithm

```
• For each permutation $u = 1, \dots, L!$
    – For each index $r = 1, \dots, L$
        ∗ The current partition is $\ell_{r,u}$
        ∗ For each particle $n = \frac{(u-1)N}{L!} + 1, \dots, \frac{uN}{L!}$
            · Obtain $\check{\mathbf{C}}^n_{\ell_{r,u},k}$ by choosing the strictest bounds available for the partition
            · Sample $\mathbf{x}^n_{\ell_{r,u},k} \sim q(\mathbf{x}_{\ell_{r,u},k}|\mathbf{x}^n_{\ell_{r,u},k-1}, \check{\mathbf{C}}^n_{\ell_{r,u},k}, \{\mathbf{x}^n_{\ell_{r',u},k},\ \ell_{r',u} : r' < r\})$
            · Compute $w^n_{\ell_{r,u},k} \propto p(\mathbf{Y}_k|\mathbf{x}^n_{\ell_{r,u}})\mathsf{N}^n_{\ell_{r,u}}$
        ∗ Normalize $w_{\ell_{r,u},k}$
        ∗ For each particle $n = \frac{(u-1)N}{L!} + 1, \dots, \frac{uN}{L!}$
            · Obtain a sample $j$ from the distribution of $w_{l,k}$ with replacement
            · Retain $\mathbf{x}^n_{\ell_{r,u},k} = \mathbf{x}^j_{\ell_{r,u},k}$, $\mathsf{N}^n_{\ell_{r,u},k} = \mathsf{N}^j_{\ell_{r,u},k}$ and $b^n_{\ell_{r,u},k} = w^j_{\ell_{r,u},k}$
            · Retain $\{\mathbf{x}^n_{\ell_{r',u},k} = \mathbf{x}^j_{\ell_{r',u},k} : r' < r\}$, $\{\mathsf{N}^n_{\ell_{r',u},k} = \mathsf{N}^j_{\ell_{r',u},k} : r' < r\}$ and $\{b^n_{\ell_{r',u},k} = w^j_{\ell_{r',u},k} : r' < r\}$
• For each particle $n = 1, \dots, N$
    – Compute $w^n_k \propto w^n_{k-1}\frac{p(\mathbf{Y}_k|\mathbf{X}^n_k)\prod_{l=1}^{L}\mathsf{N}^n_{l,k}}{\prod_{l=1}^{L} b^n_{l,k}}$
```

6.2 Kinematic Density and Measurement Model

As in the IP, the joint kinematic density is assumed to factorize into the kinematic densities of individual targets. In this case, however, we need to take into account the dependency of the targets on the group constraints and the self-constraints. Thus, we can still factorize our density with the addition of an extra term that depends on the constraints. The kinematic density for a particle n becomes

$$p(\mathbf{X}^n_k|\mathbf{X}^n_{k-1}) = \mathbf{I}(\mathbf{X}^n_k, \mathbf{C}^n_k)\prod_{l=1}^{L} p(\mathbf{x}^n_{l,k}|\mathbf{x}^n_{l,k-1}). \tag{22}$$

where $\mathbf{I}(\mathbf{X}^n_k, \mathbf{C}^n_k)$ is the particle constraint likelihood function that depends on the constraints $\mathbf{C}_k^n$. $\mathbf{C}_k^n$ represents a region where valid values of the particle state vector $\mathbf{X}_k^n$ can be found. Thus, the particle constraint likelihood function takes values as

$$\mathbf{I}(\mathbf{X}_k^n, \mathbf{C}_k^n) = \begin{cases} 1, & \text{If } \mathbf{X}_k^n \text{ is within } \mathbf{C}_k^n \\ 0, & \text{If } \mathbf{X}_k^n \text{ is outside } \mathbf{C}_k^n. \end{cases} \tag{23}$$

The details of the constraint region $\mathbf{C}_k^n$ are discussed in Section 6.3.

For a measurement model for our simulations, we will use the three angle only sensor scenario as provided in Section 5.1.2. For this model, for S sensors having I cells each, the measurement likelihood is given as a product of the likelihoods from each sensor as

$$p(\mathbf{Y}_k|\mathbf{X}_k) = \prod_{s=1}^{S} p(\mathbf{y}_{s,k}|\mathbf{X}_k) \tag{24}$$

where the likelihood relative each sensor s is obtained by the product of the likelihoods for each cell i as

$$p(\mathbf{y}_{s,k}|\mathbf{X}_k) = \prod_{i=1}^{I} p(y_{s,i,k}|\mathbf{X}_k). \tag{25}$$

6.3 Partition Ordering and Constraint Region Modeling

Concerning the variability in the order of proposing partitions, assuming that we deal with L partitions representing targets belonging in a group, the partitions can be ordered a maximum number of $L! = L(L-1)...2$ different ways. The number of orderings we choose should be much less than the number of particles used. The reason for this is that in the sampling step of the proposal, we sample several particles from each ordering. This number of particles should be sufficiently large for sampling to be useful. If we have a large number of partitions, we can reduce the number of orderings to however many we choose. A way to use as many orderings as possible, is to use a subset of the set of orderings at a certain time step and different subsets in subsequent time steps. The shorter the time intervals compared to changes in target motion, the more effective this procedure will be.

As far as constraints are concerned, we have mentioned that these are categorized as either self-constraints or group constraints. For example, a self-constraint is the restriction placed on the motion of a vehicle by the edges of a road. This restriction is independent of any other target positioning. Convoy movement imposes group constraints where the position of one target is constrained by the positions of other targets in its group.

For the different proposal orders, we can store different permutations of the indices of the targets as rows in a matrix. We proceed with the mathematical formulation of the constraints applied to each partition and the notation and organization of the different proposal orders, for a specific particle n and a time step k. To simplify the discussion, we focus on one group of targets imposing kinematic constraints on each other. The following analysis applies to each of the target index permutations.

The uth permutation, $u = 1, \dots, L!$, at time k, denoted as

$$\mathbf{d}_{u,k} = [\ell_{1,u}\ \ell_{2,u}\ \dots\ \ell_{L,u}] \tag{26}$$

consists of partition indexes $\ell_{r,u}$ where the index r runs from $1, \dots, L$. Let

$$\mathbf{D}_k = [\mathbf{d}_{1,k}\ \mathbf{d}_{2,k}\ \dots\ \mathbf{d}_{L!,k}]^T \tag{27}$$

be an L by $L!$ matrix whose rows $\mathbf{d}_{u,k}$ contain permutations of partition indexes. An example of such a matrix is given as:

$$\mathbf{D}_k = \begin{bmatrix} 1 & 2 & 3 & 4 & \dots & L \\ 1 & 3 & 2 & 4 & \dots & L \\ 1 & 3 & 4 & 2 & \dots & L \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 2 & 1 & 3 & 4 & \dots & L \\ 2 & 3 & 1 & 4 & \dots & L \\ 2 & 3 & 4 & 1 & \dots & L \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ L & L-1 & L-2 & L-3 & \dots & 1 \\ & & & & & \end{bmatrix}. \tag{28}$$

For each of these permutations, the constraints applied to each of the partitions depend on the previously proposed partitions, if any, and the self-constraints of each partition. Strictly speaking, the group constraints applied to one target need to reflect the target's dependency on all other targets in the group. However, this is not possible. This problem resembles solving one equation with many unknowns, and thus cannot yield a unique solution. Therefore, we use the following method to obtain the constraints applied to a partition: For the first target we propose, we use the self-constraints for that target, which are the only constraints available. For the second target, we use its self-constraints and the constraints due to the proposed state of the first target, and so on for the rest. Note that, only for the last proposed target we have available its self-constraints and group constraints imposed by all the state vector values of the other targets of the group. Variability in target proposal ordering will make up for the drawbacks of the necessary sequential approach in the case where, partitions that have taken values significantly different from their true values, provide misleading constraints to other targets' partitions.

For the formulation of constraints, we form regions where allowable values of state vectors lie. It is important to distinguish the constraint region associated with the entire state vector $\mathbf{X}_k^n$ and denoted as $\mathbf{C}_k^n$, from the constraint region associated with the individual partitions $\mathbf{x}_{\ell_{r,u},k}^n$ and denoted as $\check{\mathbf{C}}_{\ell_{r,u},k}^n$. The former is found after all the components of the state vector $\mathbf{X}_k^n$ have been proposed, while the latter is found during the proposal part of the COMP and CLIP.

Region $\mathbf{C}_k^n$ can be found only after we have obtained values for all state vectors of the partitions, that is after $\mathbf{X}_k^n$ is found, using the specific kinematic constraint rules of the scenario at hand. In other words $\mathbf{C}_k^n$ can be found after the complete particle has been proposed. $\mathbf{C}_k^n$ will be used in the particle weight equations of RCOMP (53) and CLIP (65) as we will see in Sections 6.5 and 7.

On the other hand, $\check{\mathbf{C}}_{\ell_{r,u},k}^n$ is used in the proposal part of the COMP and CLIP. More specifically, for the COMP $\check{\mathbf{C}}_{\ell_{r,u},k}^n$ will be used in Section 6.4 which describes the proposal process. For the CLIP in the partition weighing function (60) alone. This is provided in Section 7 . We go ahead to explain how the region $\check{\mathbf{C}}_{\ell_{r,u},k}^n$ is obtained. The constraints described by $\check{\mathbf{C}}_{\ell_{r,u},k}^n$ include both self-constraints, denoted as $\breve{\mathbf{C}}_{\ell_{r,u},k}^n$ and group constraints imposed by other partitions and denoted as $\tilde{\mathbf{C}}_{\ell_{r,u},k}^n$. As mentioned previously, the self-constraints $\breve{\mathbf{C}}_{\ell_{r,u},k}^n$ are assumed available at the time when partition $\ell_{r,u}$ is proposed, while the group constraints $\tilde{\mathbf{C}}_{\ell_{r,u},k}^n$ coming from other partitions, are formed based only on the partitions that have already been proposed. We denote the partitions that have been proposed before partition $\ell_{r,u}$ as $\ell_{r',u}$, where $r' < r$. Since the constraint region for partition $\ell_{r,u}$ is the region where values of the partition exist, then the group constraint region for the partition that is proposed first will be the set of real numbers for each of the elements of the partition.

According to the explanation above, the constraint region for $\ell_{r,u}$ is given by

$$\check{\mathbf{C}}^{n}_{\ell_{r,u},k} = \breve{\mathbf{C}}^{n}_{\ell_{r,u},k} \bigcap \tilde{\mathbf{C}}^{n}_{\ell_{r,u},k}. \tag{29}$$

The group constraints are found as

$$\tilde{\mathbf{C}}^{n}_{\ell_{r,u},k} = \bigcap \{\hat{\mathbf{C}}^{n}_{\ell_{r',u},k}\} \qquad \forall\ \ell_{r',u,k},\ \epsilon\ \{\ell_{r',u,k} : r' < r\} \tag{30}$$

where $\hat{\mathbf{C}}^{n}_{\ell_{r',u},k}$ are the constraints imposed by each previously proposed partition. In words, the vector of group constraints is the intersection of the vectors of constraints imposed by all partitions $\ell_{r',u}$. The partitions $\ell_{r',u}$ are partitions imposing constraints on $\ell_{r,u}$ that have been proposed prior to $\ell_{r,u}$.

At each time step, the strictest of all the constraints are taken. This is what the intersection of the self and group constraint vectors in (29), and the intersection of vectors making up group constraints in (30), imply.

6.4 Importance Density

Constraints in target motion can be useful information when proposing particles. Proposing particles that take into account that some values are not probable, helps avoid the undesired effects of a bad proposal. It is important to note that, since we have many degrees of freedom in choosing our importance density, we do not need to adhere strictly to these constraints, nor include all of them. It could be the case that strict adherence to all the constraints and making the allowable regions in which partitions can be proposed very small, can cause the overall performance to degrade. This can be attributed to the fact that the kinematic rules may make overconfident assertions on state vector range of values or greatly limit that range. Specifically, since constraints on a partition state vector values depend upon other proposed partitions, these are most often not so accurate as to be in position to apply strict constraints. Therefore, such constraints would bias proposals to regions where most often the true value state vector values do not exist. If the constraints are relaxed, however, other kinematic or measurement information will have more influence in the overall proposal method.

We suggest two methods of proposing particles. Both methods assume that the prior density of partitions is Gaussian. The RCOMP consists of proposing particles with a Gaussian density, and then checking whether the proposed values satisfy the constraints. If they do not, the proposal is repeated a maximum number of times. The TCOMP method is based on truncating a Gaussian density using upper and lower bounds as truncation points, and drawing values from the truncated density. We describe the steps common to both methods in Section 6.4.1, and then provide the details pertaining to each of the RCOMP and TCOMP in Sections 6.4.2 and 6.4.3 respectively. We compare the methods in Section 6.4.4.

6.4.1 Importance Density Formulation

The proposal subroutine consists of sampling from a single partition proposal and evaluating their validity with a partition weighing function. This function may be an approximation of the measurement likelihood. In this case, the use of the partition weighing function and the subsequent sampling from the resulting particle weight distribution will bias the particle proposal towards the measurements. The combination of sampling and weighing will lead to the single partition importance density. Multiplying all the single partition importance densities together, we obtain the multitarget importance density.

We proceed with the formulation of these densities. For a single partition, the proposal density requires knowledge of the constraints. The self-constraints applied in the proposal of a partition, are assumed to be known whenever required, which is the case for most realistic scenarios. For the group constraints applied due to other partitions, however, we require the knowledge of the values of the other partitions' state vectors. As we have mentioned, the problem is similar to having an equation with many unknowns, and thus, many solutions. We propose to sequentially propose partitions based on whatever information is available at that point. This sequential proposal is described mathematically as follows. For a certain permutation u and $r = 1$, we propose the first partition with index $\ell_{1,u}$ as

$$\mathbf{x}^n_{\ell_{1,u},k} \sim q(\mathbf{x}_{\ell_{1,u},k}|\mathbf{x}^n_{\ell_{1,u},k-1}, \check{\mathbf{C}}^n_{\ell_{1,u},k}) \tag{31}$$

where

$$q(\mathbf{x}_{\ell_{1,u},k}|\mathbf{x}^n_{\ell_{1,u},k-1}, \check{\mathbf{C}}^n_{\ell_{1,u},k}) = \begin{cases} \frac{p(\mathbf{x}_{\ell_{1,u},k}|\mathbf{x}^n_{\ell_{1,u},k-1})}{\mathsf{N}^n_{\ell_{1,u},k}}, & \mathbf{x}_{\ell_{1,u},k} \in \check{\mathbf{C}}^n_{\ell_{1,u},k} \\ 0, & \text{otherwise} \end{cases}$$

$\check{\mathbf{C}}^n_{\ell_{1,u},k}$ are the constraints in (29) available up to this point. These equal to the self-constraints $\breve{\mathbf{C}}^n_{\ell_{1,u},k}$ as the group constraints $\tilde{\mathbf{C}}^n_k$ are not available at this point. $\mathsf{N}^n_{\ell_{1,u},k}$ is a normalization factor equal to the integral of $p(\mathbf{x}_{\ell_{1,u},k}|\mathbf{x}^n_{\ell_{1,u},k-1})$ over the allowable region of partition values given by $\check{\mathbf{C}}^n_{\ell_{1,u},k}$. Having $\mathbf{x}^n_{\ell_{1,u},k}$ we propose the second partition with index $\ell_{2,u}$ as

$$\mathbf{x}^n_{\ell_{2,u},k} \sim q(\mathbf{x}_{\ell_{2,u},k}|\mathbf{x}^n_{\ell_{2,u},k-1}, \check{\mathbf{C}}^n_{\ell_{2,u},k}, \mathbf{x}^n_{\ell_{1,u},k}) \tag{32}$$

where $\check{\mathbf{C}}^n_{\ell_{2,u},k}$ is the intersection of the self-constraints of partition $\ell_{2,u}$ and the group constraints depending only on $\mathbf{x}^n_{\ell_{1,u},k}$. The proposal of the rest of the partitions follows in a similar fashion, and eventually for the last partition $\ell_{L,u}$ we have

$$\mathbf{x}^n_{\ell_{L,u},k} \sim q(\mathbf{x}_{\ell_{L,u},k}|\mathbf{x}_{\ell_{L,u},k-1}, \check{\mathbf{C}}^n_{\ell_{L,u},k}, \{\mathbf{x}^n_{\ell_{r,u},k}\}_{r=1}^{L-1}). \tag{33}$$

With this sequential proposal, the equation of the multitarget importance density with kinematic constraints becomes

$$\mathbf{X}^n_k \sim \prod_{r=1}^{L} q(\mathbf{x}_{\ell_{r,u},k}|\mathbf{x}^n_{\ell_{r,u},k-1}, \check{\mathbf{C}}^n_{\ell_{r,u},k}, \{\mathbf{x}^n_{\ell_{r',u},k},\ \ell^n_{r',u} : r' < r\}). \tag{34}$$

Thus, we have

$$\mathbf{X}^n_k \sim \prod_{r=1}^{L} \frac{p(\mathbf{x}_{\ell_{r,u},k}|\mathbf{x}^n_{\ell_{r,u},k-1})\mathbf{I}(\mathbf{x}^n_{\ell_{r,u},k}, \check{\mathbf{C}}^n_{\ell_{r,u},k})}{\mathsf{N}^n_{\ell_{r,u},k}}. \tag{35}$$

where $\mathsf{N}^n_{\ell_{r,u},k}$ is a normalization factor equal to the integral of $p(\mathbf{x}_{\ell_{r,u},k}|\mathbf{x}^n_{\ell_{r,u},k-1})$ over the allowable region of partition values given by $\check{\mathbf{C}}^n_{\ell_{r,u},k}$ and where the partition constraint likelihood function is defined as

$$\mathbf{I}(\mathbf{x}^n_{\ell_{r,u},k}, \check{\mathbf{C}}^n_{\ell_{r,u},k}) = \begin{cases} 1, & \text{If } \mathbf{x}^n_{\ell_{r,u},k} \text{ is within } \check{\mathbf{C}}^n_{\ell_{r,u},k} \\ 0, & \text{If } \mathbf{x}^n_{\ell_{r,u},k} \text{ is outside } \check{\mathbf{C}}^n_{\ell_{r,u},k}. \end{cases} \tag{36}$$

After drawing from the above proposal, we weigh the samples by the single target likelihood. Since there is no explicit data association, we need to use the whole measurement space for weighing each partition of a particle. This is the function in (16) that the IP and CP algorithms use. In the case of the RCOMP and TCOMP methods, however, it is beneficial to add an extra factor that reflects the influence of the constraints.

Therefore, we weigh each partition as

$$w^n_{\ell_{r,u},k} \propto p(\mathbf{Y}_k|\mathbf{x}^n_{\ell_{r,u},k}, \check{\mathbf{C}}^n_{\ell_{r,u},k}, \{\mathbf{x}^n_{\ell_{r',u},k},\ \ell_{r',u} : r' < r\}). \tag{37}$$

The exact form of the above equation will be given for each of the RCOMP and TCOMP in Sections 6.4.2 and 6.4.3 respectively.

As in the case of the IP, these weights are normalized and partitions are sampled based on the weight distribution. In the IP, however, the sampling took place from a collection of all the L partitions of a particle. Here, due to the variability in the order of proposal, the selection takes place among only partitions in the subsets of particles proposed with the same permutation. This sampling occurs independently for each different proposal order. Since we have a choice on the number of permutations we can use, and bearing in mind that as this number increases the number of partitions that each subset contains decreases, we realize that there is a tradeoff between the number of permutations and particles available in the selection step. However, as mentioned earlier, variability in the order we propose partitions can be extended to adjacent time steps. The result is the use of more permutations and more particles per subset for individual time steps.

After sampling an index j from the weight distribution $w_{\ell_{r,u},k}$ of each group, we end up with the bias for each partition $b^n_{\ell_{r,u},k} = w^j_{\ell_{r,u},k}$, where the weights $w^n_{\ell_{r,u},k}$ within each subset have been normalized. Thus, the single partition proposal density becomes:

$$q(\mathbf{x}_{\ell_{r,u},k}|\mathbf{x}^n_{\ell_{r,u},k-1}, \check{\mathbf{C}}^n_{\ell_{r,u},k}, \mathbf{x}^n_{\ell_{r',u},k}, \mathbf{Y}_k) = b^n_{\ell_{r,u},k} q(\mathbf{x}_{\ell_{r,u},k}|\mathbf{x}^n_{\ell_{r,u},k-1}, \check{\mathbf{C}}^n_{\ell_{r,u},k}, \mathbf{x}^n_{\ell_{r',u},k}),\ \{\ell_{r',u} : r' < r\}. \tag{38}$$

Since

$$q(\mathbf{X}_k|\mathbf{X}^n_{k-1}, \mathbf{Y}_k) = \prod_{r=1}^{L} q(\mathbf{x}_{\ell_{r,u},k}|\mathbf{x}^n_{\ell_{r,u},k-1}, \check{\mathbf{C}}^n_{\ell_{r,u},k}, \{\mathbf{x}^n_{\ell_{r',u},k},\ \ell_{r',u} : r' < r\}) \tag{39}$$

then the multitarget proposal becomes,

$$q(\mathbf{X}_k|\mathbf{X}^n_{k-1}, \mathbf{Y}_k) = \prod_{r=1}^{L} b^n_{\ell_{r,u},k} q(\mathbf{x}_{\ell_{r,u},k}|\mathbf{x}^n_{\ell_{r,u},k-1}, \check{\mathbf{C}}^n_{\ell_{r,u},k}, \{\mathbf{x}^n_{\ell_{r',u},k},\ \ell_{r',u} : r' < r\}). \tag{40}$$

The details of this density for the RCOMP and the TCOMP are given in Sections 6.4.2 and 6.4.3 respectively.

6.4.2 Reproposed Constrained Motion Proposal Algorithm (RCOMP)

In this method, for each particle n, we propose each element of each partition $\mathbf{x}^n_{\ell_{r,u},k}$ using a regular Gaussian, which is the kinematic prior $p(\mathbf{x}_{\ell_{r,u},k}|\mathbf{x}^n_{\ell_{r,u},k-1})$. Therefore, for the x

position coordinate of the partition,

$$x^n_{\ell_{r,u},k} \sim \mathcal{N}(x^n_{\ell_{r,u},k-1}, \sigma^2_x) \tag{41}$$

where σ^2_x is the process noise for the x position coordinate. Then, a simple loop will continue to sample from the Gaussian proposal until the value of the nth particle $x^n_{\ell_{r,u},k}$ satisfies the constraints, that is, falls within the constraint region $\check{\mathbf{C}}^n_{\ell_{r,u},k}$ or a maximum number of samples is exceeded. We proceed similarly with the rest of the elements of $\mathbf{x}^n_{\ell_{r,u},k}$. If the constraints involve many elements simultaneously, we may propose all elements involved and then check this result with the constraints.

Since the repeated Gaussian may fail to propose partitions inside the constraints, if it reaches the maximum allowable number of tries, we need to remove the partitions that lie inside the constraints. Otherwise, in the final resampling step, if partitions that are outside the constraints exist, they will nullify the weights for particles that may have other partitions that are good. The partition constraint likelihood function $\mathbf{I}(\mathbf{x}^n_{\ell_{r,u},k}, \check{\mathbf{C}}^n_{\ell_{r,u},k})$ in (36) will reduce the single partition weights in the proposal to zero and, thus, particles conforming to the constraints, will be preferred at the subsequent sampling step. Therefore, we weigh each partition as

$$w^n_{\ell_{r,u},k} \propto p(\mathbf{Y}_k|\mathbf{x}^n_{\ell_{r,u},k})\mathbf{I}(\mathbf{x}^n_{\ell_{r,u},k}, \check{\mathbf{C}}^n_{\ell_{r,u},k}). \tag{42}$$

Using (40) and the fact that we propose from $p(\mathbf{x}_{\ell_{r,u},k}|\mathbf{x}^n_{\ell_{r,u},k-1})$, we obtain the multitarget proposal density specific to the RCOMP as:

$$q(\mathbf{X}_k|\mathbf{X}^n_{k-1}, \mathbf{Y}_k) = \prod_{r=1}^{L} b^n_{\ell_{r,u},k} p(\mathbf{x}_{\ell_{r,u},k}|\mathbf{x}^n_{\ell_{r,u},k-1})\mathbf{I}(\mathbf{x}^n_{\ell_{r,u},k}, \check{\mathbf{C}}^n_{\ell_{r,u},k}). \tag{43}$$

6.4.3 Truncated Gaussian Constrained Motion Proposal Algorithm (TCOMP)

Here, we provide the details of the TCOMP.

For each particle n, we propose each element of each partition $\mathbf{x}^n_{\ell_{r,u},k}$ using a truncated version of the kinematic prior $p(\mathbf{x}_{\ell_{r,u},k}|\mathbf{x}^n_{\ell_{r,u},k-1})$. This will be a truncated Gaussian distribution. Therefore, for the x position coordinate of the partition,

$$x_{\ell_{r,u},k} \sim \mathcal{TN}(\mu_{\ell_{r,u},k}, \sigma^2_x, x^{\text{LB}}_{\ell_{r,u},k}, x^{\text{UB}}_{\ell_{r,u},k}) \tag{44}$$

where σ^2_x is the process noise for the x position coordinate, $x^{\text{LB}}_{\ell_{r,u},k}$ a lower bound and $x^{\text{UB}}_{\ell_{r,u},k}$

an upper bound. Thus, the truncated Gaussian is given by

$$f_x^{TN}(x_{\ell_{r,u},k}) = \frac{p(\mathbf{x}_{\ell_{r,u},k}|\mathbf{x}^n_{\ell_{r,u},k-1})}{\Phi(\frac{x^{\text{UB}}_{\ell_{r,u},k}-\mu_{\ell_{r,u},k}}{\sigma_x^2}) - \Phi(\frac{x^{\text{LB}}_{\ell_{r,u},k}-\mu_{\ell_{r,u},k}}{\sigma_x^2})} I(x_{\ell_{r,u},k}) \tag{45}$$

where $\Phi(.)$ is the standard normal cumulative distribution function [52] and $I(x_{\ell_{r,u},k})$ an indicator function taking values as

$$I(x_{\ell_{r,u},k}) = \begin{cases} 1, & \text{If } x^{\text{LB}}_{\ell_{r,u},k} \leq x_{\ell_{r,u},k} \leq x^{\text{UB}}_{\ell_{r,u},k} \\ 0, & \text{Otherwise.} \end{cases}$$

We proceed similarly for the rest of the dimensions of the state vector. We will denote the normalization factor for position coordinate $x_{\ell_{r,u},k}$ and particle n as

$$\mathsf{N}^n_{x,\ell_{r,u},k} = \Phi(\frac{x^{\text{UB}}_{\ell_{r,u},k} - \mu_{\ell_{r,u},k}}{\sigma_x^2}) - \Phi(\frac{x^{\text{LB}}_{\ell_{r,u},k} - \mu_{\ell_{r,u},k}}{\sigma_x^2}). \tag{46}$$

and similarly for the rest of the dimensions. Therefore, the normalization factor for a partition becomes

$$\mathsf{N}^n_{\ell_{r,u},k} = \mathsf{N}^n_{x,\ell_{r,u},k}\mathsf{N}^n_{\dot{x},\ell_{r,u},k}\mathsf{N}^n_{y,\ell_{r,u},k}\mathsf{N}^n_{\dot{y},\ell_{r,u},k}, \tag{47}$$

where we take into account all the normalization factors of all the state vector dimensions.

According to the above, the single target importance density for partition $\ell_{r,u}$ becomes

$$q(\mathbf{x}_{\ell_{r,u},k}|\mathbf{x}^n_{\ell_{r,u},k-1}, \check{\mathbf{C}}^n_{\ell_{r,u},k}, \{\mathbf{x}^n_{\ell_{r',u},k},\ l_{r',u} : r' < r\}) = \frac{p(\mathbf{x}_{\ell_{r,u},k}|\mathbf{x}^n_{\ell_{r,u},k-1})}{\mathsf{N}^n_{\ell_{r,u},k}}. \tag{48}$$

We furthermore need to take into account the measurements. In the truncated Gaussian case, all the partition values are brought within the constraints. However, as we have mentioned earlier, partitions proposed with a mean value that is far from the constraints, will suffer from near to zero normalization factors $\mathsf{N}^n_{\ell_{r,u},k}$. These will appear at the numerator of the final weight equation, giving near zero weights to particles that may have other good partitions. Therefore, an effort to remove those partitions at the proposal subroutine will give way to partitions that are already closer to constraints if there exist any. We choose to use:

$$w^n_{\ell_{r,u},k} \propto p(\mathbf{Y}_k|\mathbf{x}^n_{\ell_{r,u},k})\mathsf{N}^n_{\ell_{r,u},k}. \tag{49}$$

Using (40) and (48) the multitarget proposal becomes

$$q(\mathbf{X}_k|\mathbf{X}_{k-1}^n, \mathbf{Y}_k) = \prod_{r=1}^{L} \frac{b_{\ell_{r,u},k}^n p(\mathbf{x}_{\ell_{r,u},k}|\mathbf{x}_{\ell_{r,u},k-1}^n)\mathbf{I}(\mathbf{x}_{\ell_{r,u},k}^n, \check{\mathbf{C}}_{\ell_{r,u},k}^n)}{\mathsf{N}_{\ell_{r,u},k}^n}. \tag{50}$$

where $\mathbf{I}(\mathbf{x}_{\ell_{r,u},k}^n, \check{\mathbf{C}}_{\ell_{r,u},k}^n)$ as in (36).

6.4.4 Comparison Between RCOMP and TCOMP

Comparing the two proposed methods, we can infer that if the constraints are correct, the TCOMP method will propose partitions with erroneous values in the time step $k-1$, to the right place in time step k. The RCOMP method will have a harder time doing so even with a large number of repetitions. This is because, the mean of the Gaussian distribution used for the proposal of the partition is then far from the partitions true value and, thus, from the permissible region defined constraints. To remedy this, in the case of the RCOMP, the partition constraint likelihood function $\mathbf{I}(\mathbf{x}_{\ell_{r,u,k}}^n, \check{\mathbf{C}}_{\ell_{r,u},k}^n)$ in is placed on the numerator of the partition weighing function as described in this section. Moreover, the function $\mathbf{I}(\mathbf{X}_k^n, \mathbf{C}_k^n)$ in (23) inherently exists in the final weighing step. The application of these functions eliminates the effect of partitions and particles not conforming with the rules on the joint posterior.

A subtle point, however, concerning the truncated Gaussian case, is that the factor $\mathsf{N}_{\ell_{r,u},k}^n$ which is necessary to appear in the numerator of the weight equation will be nearly zero for partitions that have taken highly erroneous values. This can easily be seen by observing that, for an erroneous partition, the upper and lower constraints will be placed on the tail of the kinematic distribution and that the normalization factor will be the area under the graph bounded by the constraints. This will tend to zero as the constraints move away from the mean. Therefore, highly erroneous partitions that have been proposed in right places in a subsequent step, will receive near to zero weights. However, the problem of lost tracks is still alleviated by the truncated Gaussian method, as it allows less particles to drift away from their correct values, thus improving tracking performance.

In practice, it was found that the main drawback of the TCOMP is that it requires upper and lower bounds as constraints in each position coordinate. If the constraints, however, depend on more than one position coordinates of the state vector simultaneously, the TCOMP is hard to implement. The RCOMP, however, as described above, handles this situation easily.

6.5 Computation of Particle Weights

For the final weighing of the complete particles, we need to take into account all the functions used for the proposal. Here we use the index l, $l = 1, \ldots, L$ to denote the partitions since the proposal order does not matter. The weight of the nth particle, $n = 1, \ldots, N$, will be

$$w_k^n \propto w_{k-1}^n \frac{p(\mathbf{Y}_k|\mathbf{X}_k^n)p(\mathbf{X}_k^n|\mathbf{X}_{k-1}^n)}{q(\mathbf{X}_k^n|\mathbf{X}_{k-1}^n, \mathbf{Y}_k)}. \tag{51}$$

For the RCOMP, using (43) we have

$$w_k^n \propto w_{k-1}^n \frac{p(\mathbf{Y}_k|\mathbf{X}_k^n)p(\mathbf{X}_k^n|\mathbf{X}_{k-1}^n)}{\prod_{l=1}^L b_{l,k}^n p(\mathbf{x}_{l,k}^n|\mathbf{x}_{l,k-1}^n)} \tag{52}$$

which after simplification results to

$$w_k^n \propto w_{k-1}^n \frac{p(\mathbf{Y}_k|\mathbf{X}_k^n)\mathbf{I}(\mathbf{X}_k^n, \mathbf{C}_k^n)}{\prod_{l=1}^L b_{l,k}^n} \tag{53}$$

and for the TCOMP, using (50) we have

$$w_k^n \propto w_{k-1}^n \frac{p(\mathbf{Y}_k|\mathbf{X}_k^n)p(\mathbf{X}_k|\mathbf{X}_{k-1}^n)}{\prod_{l=1}^L \frac{b_{l,k}^n p(\mathbf{x}_{l,k}|\mathbf{x}_{l,k-1}^n)}{\mathsf{N}_{l,k}^n}} \tag{54}$$

which after simplification results to

$$w_k^n \propto w_{k-1}^n \frac{p(\mathbf{Y}_k|\mathbf{X}_k^n)\prod_{l=1}^L \mathsf{N}_{l,k}^n}{\prod_{l=1}^L b_{l,k}^n}. \tag{55}$$

6.6 Example of Particle Construction with the COMP

Here, we provide a two target example for which we have two ways of changing the order of proposing partitions. The example demonstrates the construction of particles for the two subsets, each having a different order of proposing partitions. More specifically, out of the complete set of particles, we propose the first subset, including half the particles, using the order given by the first permutation, $u = 1$ (partition $\mathbf{x}_{\ell_{1,1}}^n = \mathbf{x}_1^n$, then partition $\mathbf{x}_{\ell_{2,1}}^n = \mathbf{x}_2^n$), and the second subset having the other half of the particles with the order given by the second permutation (partition $\mathbf{x}_{\ell_{1,2}}^n = \mathbf{x}_2^n$, then partition $\mathbf{x}_{\ell_{2,2}}^n = \mathbf{x}_1^n$). This occurs at each time step k, therefore the subscript k has been omitted for simplicity.

In Table 8, in Step (i), for the proposal order given by the first permutation, $u = 1$, we propose partition $\mathbf{x}_1^n$ for $n = 1, \ldots, 5$. We weigh and sample from the population of partition

Table 8: Particle Construction with the COMP

$u = 1$								
$\{\mathbf{x}_1^1\}$		$\{\mathbf{x}_1^2\}$	$\rightarrow$	$\{\mathbf{x}_2^1\}$	$\Rightarrow$	$\{\mathbf{x}_1^2, \mathbf{x}_2^1\}$		$\{\mathbf{x}_1^3, \mathbf{x}_2^2\}$
$\{\mathbf{x}_1^2\}$		$\{\mathbf{x}_1^3\}$	$\rightarrow$	$\{\mathbf{x}_2^2\}$	$\Rightarrow$	$\{\mathbf{x}_1^3, \mathbf{x}_2^2\}$		$\{\mathbf{x}_1^1, \mathbf{x}_2^5\}$
$\{\mathbf{x}_1^3\}$	$\xrightarrow{\text{sample } \mathbf{x}_1^n}$	$\{\mathbf{x}_1^3\}$	$\rightarrow$	$\{\mathbf{x}_2^3\}$	$\Rightarrow$	$\{\mathbf{x}_1^3, \mathbf{x}_2^3\}$	$\xrightarrow{\text{sample } \mathbf{x}_2^n}$	$\{\mathbf{x}_1^2, \mathbf{x}_2^1\}$
$\{\mathbf{x}_1^4\}$		$\{\mathbf{x}_1^5\}$	$\rightarrow$	$\{\mathbf{x}_2^4\}$	$\Rightarrow$	$\{\mathbf{x}_1^5, \mathbf{x}_2^4\}$		$\{\mathbf{x}_1^3, \mathbf{x}_2^2\}$
$\{\mathbf{x}_1^5\}$		$\{\mathbf{x}_1^1\}$	$\rightarrow$	$\{\mathbf{x}_2^5\}$	$\Rightarrow$	$\{\mathbf{x}_1^1, \mathbf{x}_2^5\}$		$\{\mathbf{x}_1^1, \mathbf{x}_2^5\}$
(i)		(ii)		(iii)		(iv)		(v)
$u = 2$								
$\{\mathbf{x}_2^1\}$		$\{\mathbf{x}_2^3\}$	$\rightarrow$	$\{\mathbf{x}_1^1\}$	$\Rightarrow$	$\{\mathbf{x}_1^1, \mathbf{x}_2^3\}$		$\{\mathbf{x}_1^2, \mathbf{x}_2^5\}$
$\{\mathbf{x}_2^2\}$		$\{\mathbf{x}_2^5\}$	$\rightarrow$	$\{\mathbf{x}_1^2\}$	$\Rightarrow$	$\{\mathbf{x}_1^2, \mathbf{x}_2^5\}$		$\{\mathbf{x}_1^3, \mathbf{x}_2^5\}$
$\{\mathbf{x}_2^3\}$	$\xrightarrow{\text{sample } \mathbf{x}_2^n}$	$\{\mathbf{x}_2^5\}$	$\rightarrow$	$\{\mathbf{x}_1^3\}$	$\Rightarrow$	$\{\mathbf{x}_1^3, \mathbf{x}_2^5\}$	$\xrightarrow{\text{sample } \mathbf{x}_1^n}$	$\{\mathbf{x}_1^3, \mathbf{x}_2^5\}$
$\{\mathbf{x}_2^4\}$		$\{\mathbf{x}_2^4\}$	$\rightarrow$	$\{\mathbf{x}_1^4\}$	$\Rightarrow$	$\{\mathbf{x}_1^4, \mathbf{x}_2^4\}$		$\{\mathbf{x}_1^5, \mathbf{x}_2^1\}$
$\{\mathbf{x}_2^5\}$		$\{\mathbf{x}_2^1\}$	$\rightarrow$	$\{\mathbf{x}_1^5\}$	$\Rightarrow$	$\{\mathbf{x}_1^5, \mathbf{x}_2^1\}$		$\{\mathbf{x}_1^4, \mathbf{x}_2^4\}$

$\mathbf{x}_1^n$ and result in the sampled version shown in Step (ii). At Step (iii), we propose partitions $\mathbf{x}_2^n$ based on constraints provided by partitions $\mathbf{x}_1^n$. This results in the augmented particles of Step (iv). In Step (v), we weigh and sample from that population of particles, based on the weight distribution that partition $\mathbf{x}_2^n$ has. Here, the weights of the particles of Step (iv) have been obtained based on the validity of partitions $\mathbf{x}_2^n$ only and not the combination of $\mathbf{x}_1^n$ and $\mathbf{x}_2^n$. Its is also worth mentioning that if a sample of $\mathbf{x}_1^n$ has survived the sampling from steps (i) to (ii), it may not survive the sampling from steps (iv) to (v) if a sample of $\mathbf{x}_2^n$ whose proposal was biased by the constraints of $\mathbf{x}_1^n$, is not good. For example, while $\mathbf{x}_1^5$ has survived the first sampling, it fails to appear among the selection of the second sampling as the sample $\mathbf{x}_2^4$ (whose proposal it affected with its constraints) does not obtain a good weight. For the second permutation, the same steps are taken but proposing first $\mathbf{x}_2^n$ then $\mathbf{x}_1^n$. When we have more targets, we have more partitions for each particle and, therefore, we can have more permutations. In this case, the procedure for obtaining all particles is an obvious extension of the procedure outlined above, by adding more steps and permutations as the number of partitions increases.

6.7 Algorithm Description

The detailed steps of the COMP procedure are as follows. They are also summarized in Table 6 for the RCOMP and in Table 7 for the TCOMP.

- We use a motion model with the resulting density:

$$p(\mathbf{X}_k^n|\mathbf{X}_{k-1}^n) = \mathbf{I}(\mathbf{X}_k^n, \mathbf{C}_k^n) \prod_{l=1}^{L} p(\mathbf{x}_{l,k}^n|\mathbf{x}_{l,k-1}^n). \tag{56}$$

- We also use a measurement model with the resulting density:

$$p(\mathbf{Y}_k|\mathbf{X}_k^n). \tag{57}$$

  These two models are presented in Section 6.2, and are used in the evaluation of particles in Section 6.5.

- We have L partitions indexed by $l = 1, \dots, L$; each partition corresponds to one of L assumed targets.

- By finding all permutations of these partitions, we form the matrix $\mathbf{D}_k$ in (28), having $L!$ rows and L columns. Each row is a permutation of the partition indices.

- We decide on how many of the $L!$ permutations to use (see Section 6.3). For the following description of the algorithm we choose to use all $L!$ permutations.

- We divide the set of N particles into $L!$ subsets. Each subset is composed of $\frac{N}{L!}$ particles. (For convenience we use N as a multiple of $L!$). Therefore, the first subset has particles with indices $1, \dots \frac{N}{L!}$, the second subset has particles with indices $\frac{N}{L!} + 1, \dots \frac{2N}{L!}$, ...and the $L!$th subset particles with indices $\frac{(L!-1)N}{L!} + 1, \dots N$.

- To propose the first subset of particles ($n = 1, \dots \frac{N}{L!}$), we use the order of the partitions given by the first row $\mathbf{d}_{1,k}$ of the permutation matrix $\mathbf{D}_k$ in (28). This order is $\ell_{r,1} = 1, 2, \dots, L$.(see Section 6.3).
  - Propose realizations of the partition assigned to be proposed first in this subset of particles. This is partition $\ell_{1,1} = 1$.
    - For the each particle in this subset ($n = 1$ to $\frac{N}{L!}$):
      - Obtain the region of constraints $\check{\mathbf{C}}^n_{1,k}$ as described in Section 6.3.
      - Sample from a single target importance density corresponding to the nth partition: $\mathbf{x}^n_{1,k} \sim q(\mathbf{x}_{1,k}|\mathbf{x}^n_{1,k-1}, \check{\mathbf{C}}^n_{1,k})$ (Section 6.4).
      - In the case of the RCOMP: if the constraints have not been met, return to previous step; quit after a number of tries.
      - Compute the weight $w^n_{1,k}$ as described in section 6.5.
    - Normalize $w_{1,k}$. Now we have a distribution of weights each corresponding to a realization of partition $\ell_{1,1} = 1$. If we sample from this distribution with replacement, it is more likely to select the samples having the biggest weights.
    - For each particle in this subset ($n = 1$ to $\frac{N}{L!}$):
      - Obtain a sample j from the distribution of $w_{1,k}$ with replacement
      - For the RCOMP: Retain $\mathbf{x}^1_{1,k} = \mathbf{x}^j_{1,k}$ for estimation and $b^1_{1,k} = w^j_{1,k}$ for use in the particle weighing in (53).
      - For the TCOMP: Retain $\mathbf{x}^1_{1,k} = \mathbf{x}^j_{1,k}$ for estimation and $\mathsf{N}^1_{1,k} = \mathsf{N}^j_{1,k}$ and $b^1_{1,k} = w^j_{1,k}$ for use in the particle weighing in (55).

- The procedure is repeated with the same steps as for partition $\ell_{1,1} = 1$, with the difference that after obtaining a sample j from the distribution of $w_{2,k}$ with replacement, we add the following steps:
  - For the RCOMP: Retain $\{\mathbf{x}^n_{1,k} = \mathbf{x}^j_{1,k}\}$ and $\{b^n_{1,k} = w^j_{1,k}\}$. This is due to the fact that $\ell^1_{2,1,k} = 2$ is coupled with $\ell^1_{1,u,k} = 1$. These two should not be separated. For more detail see Section 6.6.
  - For the TCOMP: Retain $\{\mathbf{x}^n_{1,k} = \mathbf{x}^j_{1,k}\}$, $\{\mathsf{N}^n_{1,k} = \mathsf{N}^j_{1,k}\}$ and $\{b^n_{1,k} = w^j_{1,k}\}$ for the same reasons as in RCOMP.

- Repeat according to the rest of the rows of the permutation matrix until all subsets of particles have been proposed.

- For each particle $n = 1, \ldots, N$

  - For the RCOMP perform the particle weighing step by computing: $w_k^n \propto w_{k-1}^n \frac{p(\mathbf{Y}_k|\mathbf{X}_k^n)\mathbf{I}(\mathbf{X}_k^n,\mathbf{C}_k^n}{\prod_{l=1}^L b_{l,k}^n}$
  - For the TCOMP perform the particle weighing step by computing: $w_k^n \propto w_{k-1}^n \frac{p(\mathbf{Y}_k|\mathbf{X}_k^n)\prod_{l=1}^L \mathsf{N}_l^n}{\prod_{l=1}^L b_{l,k}^n}$

7 Constraint Likelihood Function Independent Partition Algorithm

Due to the algorithmic and computational complexity associated with the COMP algorithm proposed in the previous section, we consider a simplified algorithm, the constraint likelihood function independent partition method (CLIP), which still makes use of information provided by constrained motion via the constraint likelihood function alone. This method is simpler and less computationally expensive. The main motivation of considering a simplified method is the fact that sometimes more naive proposals can reach the level of performance of more sophisticated methods, merely by increasing the number of particles used. In general, a simplified approach will be less computationally expensive than a more sophisticated one, for the same number of particles. Therefore, we can consider increasing the number of particles of a simpler method to match the execution time of the more complex method and make a fair comparison of the two methods at similar levels of execution time.

The method we propose is basically the IP method but with partition constraint likelihood functions used in the partition weighing function and with the particle constraint likelihood function used in the final weighing equation. These functions are exactly the ones used in the RCOMP method. The differences of the CLIP method to the RCOMP method is that the CLIP does not use variability in proposing partitions and does not repropose partitions to satisfy the constraints. Therefore, the computational expense is greatly reduced in the CLIP versus both the RCOMP and TCOMP methods.

Next, we provide the equations used by the CLIP. We use the same motion model as in the case of the COMP in (22):

$$p(\mathbf{X}_k^n|\mathbf{X}_{k-1}^n) = \mathbf{I}(\mathbf{X}_k^n, \mathbf{C}_k^n)\prod_{l=1}^{L} p(\mathbf{x}_{l,k}^n|\mathbf{x}_{l,k-1}^n). \tag{58}$$

where $\mathbf{I}(\mathbf{X}_k^n, \mathbf{C}_k^n)$ as defined in (23) and $\mathbf{C}_k^n$ as described in Section 6.3. Moreover, it is convenient to sample from the kinematic prior for each partition l as

$$\mathbf{x}_{l,k}^n \sim p(\mathbf{x}_{l,k}|\mathbf{x}_{l,k-1}^n) \tag{59}$$

.

Apart from sampling from the kinematic prior, we use a particle weight function. We include measurement information by using the partition likelihood function, $p(\mathbf{Y}_k|\mathbf{x}_{l,k}^n)$ and constraint information using the partition constraint likelihood function, $\mathbf{I}(\mathbf{x}_{l,k}^n, \check{\mathbf{C}}_{l,k}^n)$, where $\check{\mathbf{C}}_{l,k}^n$ as in (29), for $l = 1, \ldots, L$. Therefore, the partition weight function is given by

$$w_{l,k}^n \propto p(\mathbf{Y}_k|\mathbf{x}_{l,k}^n)\mathbf{I}(\mathbf{x}_{l,k}^n, \check{\mathbf{C}}_{l,k}^n). \tag{60}$$

After resampling from the distribution of $w_{l,k}$, the weight $w_{l,k}^j$ received by each partition $l = 1, \ldots, L$ becomes the bias $b_{l,k}^n$ for that partition, corresponding to particle n. Therefore, $b_{l,k}^n = w_{l,k}^j$.

Moreover, from (51), the weight of the particle $n = 1, \ldots, N$ is:

$$w_k^n \propto w_{k-1}^n \frac{p(\mathbf{Y}_k|\mathbf{X}_k^n)p(\mathbf{X}_k^n|\mathbf{X}_{k-1}^n)}{q(\mathbf{X}_k^n|\mathbf{X}_{k-1}^n, \mathbf{Y}_k)} \tag{61}$$

where $q(\mathbf{X}_k^n|\mathbf{X}_{k-1}^n, \mathbf{Y}_k)$ is the proposal density used to propose each of the particles. We recall that each of the partitions of the particle was proposed by the kinematic prior of the target it represents and weighted with the whole posterior and the constraint likelihood. This weight biased its selection for placement in the particle. Therefore,

$$q(\mathbf{X}_k^n|\mathbf{X}_{k-1}^n, \mathbf{Y}_k) = \prod_{l=1}^{L} q(\mathbf{x}_{l,k}^n|\mathbf{x}_{l,k-1}^n, \mathbf{Y}_k) \tag{62}$$

and

$$q(\mathbf{x}_{l,k}^n|\mathbf{x}_{l,k-1}^n, \mathbf{Y}_k) = b_{l,k}^n p(\mathbf{x}_{l,k}^n|\mathbf{x}_{l,k-1}^n). \tag{63}$$

Combining (62) and (63), the weight of each particle in (61) becomes

$$w_k^n \propto w_{k-1}^n \frac{p(\mathbf{Y}_k|\mathbf{X}_k^n)p(\mathbf{X}_k^n|\mathbf{X}_{k-1}^n)}{(\prod_{l=1}^{L} b_{l,k}^n)(\prod_{l=1}^{L} p(\mathbf{x}_{l,k}^n|\mathbf{x}_{l,k-1}^n))} \tag{64}$$

which using (58) simplifies as

Table 9: Constraint Independent Partitions Algorithm

```
• For each partition $l = 1, \ldots, L$
    – For each particle $n = 1, \ldots, N$
        ∗ Sample $\mathbf{x}_{l,k}^n \sim p(\mathbf{x}_{l,k}|\mathbf{x}_{l,k-1}^n)$
        ∗ Compute $w_{l,k}^n \propto p(\mathbf{Y}_k|\mathbf{x}_{l,k}^n)\mathbf{I}(\mathbf{x}_{l,k}^n, \check{\mathbf{C}}_{l,k}^n)$
    – Normalize $w_{l,k}$
    – Resample by choosing a partition $j$ from the distribution of $w_{l,k}$ with replacement
    – Keep the bias of each sample as $b_{l,k}^n = w_{l,k}^j$
• For each particle $n = 1, \ldots, N$
    – Compute $w_k^n \propto w_{k-1}^n \frac{p(\mathbf{Y}_k|\mathbf{X}_k^n)\mathbf{I}(\mathbf{X}_k^n, \mathbf{C}_k^n)}{\prod_{l=1}^L b_{l,k}^n}$.
```

$$w_k^n \propto w_{k-1}^n \frac{p(\mathbf{Y}_k|\mathbf{X}_k^n)\mathbf{I}(\mathbf{X}_k^n, \mathbf{C}_k^n)}{\prod_{l=1}^L b_{l,k}^n}. \tag{65}$$

The algorithm is provided in Table 9.

8 Simulation Results

We have chosen two military applications that provide suitable conditions for assessing the performance of the algorithms proposed in this report. One of them considers military vehicles moving in convoy motion [15], and the other application is the common leapfrog motion [16] that military vehicles or personnel employ to achieve various goals.

8.1 Convoy Motion

In convoy motion, military vehicles move as groups in order to execute a certain task effectively. Convoy motion is mainly employed for reasons of safety. It is important for the individual elements of the convoy to follow certain rules in their motion. The speed of the convoy and the separation among the various elements are determined by specific organization requirements. Therefore, the knowledge that a certain group of vehicles moves

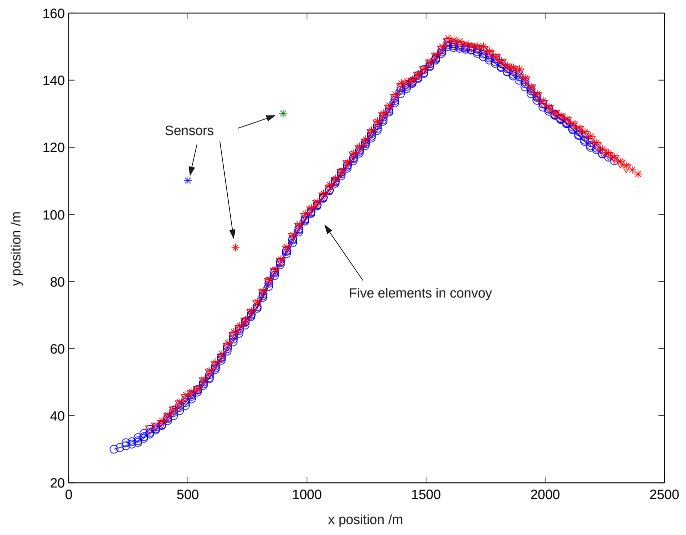

Figure 6: Five targets moving one behind the other throughout the entire motion in the scenario. The targets move from the left hand side to the right and from bottom to top

in a convoy, can be very beneficial to the tracker.

In our convoy motion scenario, there are five military vehicles that move one behind the other as in Figure 6. The individual vehicles move with an average speed of 90 km/h and keep a distance of about 50 m apart. The motion consists of 81 time steps. The tracker is assumed to know the number of targets and their approximate separation distance. The order in which the vehicles move throughout the scenario is also assumed unchanged. The tracker implements the kinematic rules by not allowing position estimates of adjacent targets to be placed any closer than 40 m and further than 60 m apart. The targets use a constant motion model with a dynamic noise covariance given by the diagonal matrix Q in (9) with diagonal entries [20 .5 20 .5]. The measurement model described in Section 5.1.2 is used with each of the cells of the three angle-only sensors covering an angle of pi/64 radians. The probability of false alarm is set to 10^{-2} and the SNR, denoted as λ in (11), to 20 db. The sensors are positioned as shown in Figure 6 where the motion of the five targets is also shown. The tracker proposes particles by dividing them in $4! = 24$ subsets for reasons explained in 6.3. As the execution time of the algorithms is directly related to the number of particles we use, we vary the number of particles, thus the execution time, and compare the root mean square error (RMSE) performance of the IP, CLIP and RCOMP algorithm versus the program execution time, obtained on a Pendium 4, 3 GHz, with 1 Gb

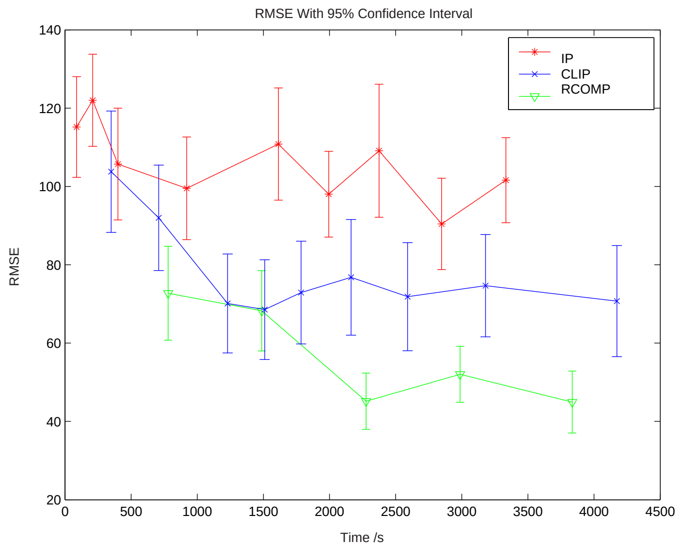

Figure 7: RMSE vs the time that the simulation takes to complete 81 time steps.

of memory, with all factors affecting computational time kept the same for all methods to the best of our knowledge. This is to show that even though the use of constraint motion information can be computationally expensive, it is worthwhile to employ it as it reaches levels of performance unmatched by the IP.

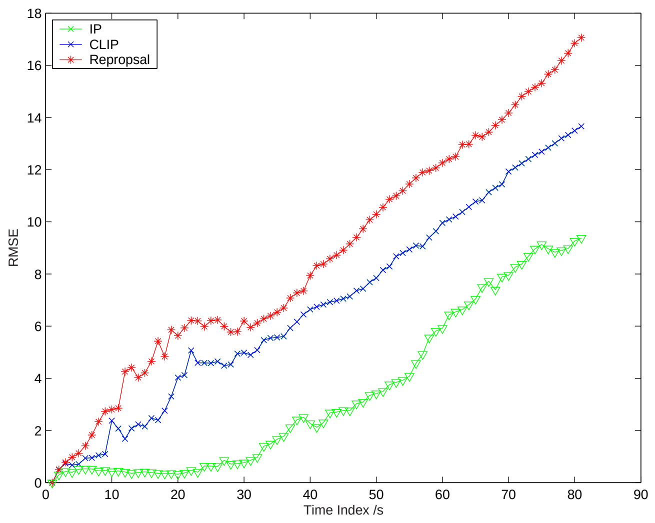

Figure 8: RMSE vs the time step for the entire motion of the targets.

From Figure 7 we can see that even if we increase the number of particles for the IP, thus making it more computationally expensive and reducing the RMSE, it can never match the performance of the RCOMP. Therefore, we reach the conclusion that to obtain the best performance in such a scenario, the information about kinematic constraints that is available, has to be used. It needs, however, to be used in an effective way. The CLIP, that is less computationally expensive than the RCOMP still fails to reach the levels of performance of the RCOMP due to its ineffective use of the information provided. This is true even if the number of particles used by CLIP is increased. Furthermore, in the same graph we plot the 95% confidence intervals around the mean of each RMSE estimate. This is an indication of the reliability of the scheme used as the confidence intervals are directly related to the variace of the estimates. Clearly, the RCOMP is the most reliable scheme with the lowest variance. It should be noted that a comparison between the schemes is only meaningful at a certain execution time and beyond, where the RMSE reaches a steady state. For small execution times, we have too few particles for any particle filtering algorithm to perform well.

In Figure 8, the RMSE performance of the IP, CLIP and RCOMP are plotted versus the time index corresponding to each time step in the scenario. The numbers of particles that were taken are the ones that result in almost the same computational expense, around 4000s, for all the algorithms, in order to make the comparison fair. One observation is that the RMSE increases as the scenario time progresses. This is due to the fact that, in the beginning, we initialize our tracker with estimates that are close to the true ones, thus obtaining a small estimation error. Furthermore, as the time progresses, the targets move away from the sensors. Since we use sensor cells that each observe an angle of our observation region, the further we move from the sensors, the more the resolution of our observation cells decreases. Here, the COMP is found to have a lower RMSE compared to the corresponding values of the RMSE for the CLIP and IP for the same time index. The CLIP appears to have a slight advantage over the IP.

8.2 Leapfrog

Leapfrog is a commonly used technique in the military [16]. It consists of having groups of personnel, vehicles or aircrafts move in a desired direction, while one or more other groups remain stationary, providing cover to the moving group by suppressing enemy fire. Timing and precision in the execution of this kind of motion is crucial. It provides effective cover against enemy fire and helps the group in motion avoid friendly fire. A tracker can utilize information coming from this kinematic dependence of the targets and improve its performance in difficult scenarios.

In the leapfrog scenario used in this report, we track the motion of four soldiers performing leapfrog motion in groups of two. The soldiers attempt to advance closer to enemy lines with as much rapidity and cover as possible. Therefore, at each segment of their motion, two soldiers remain stationary at fortified positions or objects that obstruct enemy vision and fire, providing cover to the other two soldiers that advance forward. The advancing soldiers run with a zigzag fashion, in order to avoid being near stationary targets in the direction facing the enemy. They also try not to deviate too much from their mean position to their left or right. This is necessary as friendly fire exists on their sides. It is also important for the moving soldiers to advance forward by the same amount. The various stages of this motion are described in Figures 10, 11,12,13 and 14. The information that the target has in this simulation is the constant number of targets and the grouping of the targets. The tracker assumes that the moving targets are not to move in front of the stationary targets, a constraint in the x position coordinate, and that the moving targets advance forward by approximately the same amount, a constraint in the y position coordinate. More specifically, for the x position coordinate, estimates of any target are not to be placed any closer than 4 meters to any other target's estimate. Moreover, for the targets of the same group, the predicted values of the y position coordinate of their state vector should not differ by 2 meters. The motion consists of 81 time steps. A constant velocity model with constraints is applied in this scenario having a dynamic noise dynamic noise covariance given by the diagonal matrix Q in (9) with diagonal entries $Q = [.07\ .001\ .4\ 1]$. The same measurement model as in the convoy scenario is used with each of the cells of our angle-only sensors covering an angle of pi/128 radians. The positioning of the sensors, together with the complete motion of the targets is given in Figure 9. The tracker proposes particles by dividing them in 24 subsets. We compare the root mean square error (RMSE) performance of the IP, CLIP, RCOMP and TCOMP versus the program execution time for the same reasons as in the convoy scenario. We also provide RMSE versus the time index, for an execution time of about 3000s, corresponding to each of the 81 time steps of the scenario.

In Figure 15 we can see that, similarly to the convoy scenario case, even if we increase the number of particles for the IP, it can never match the performance of the RCOMP and

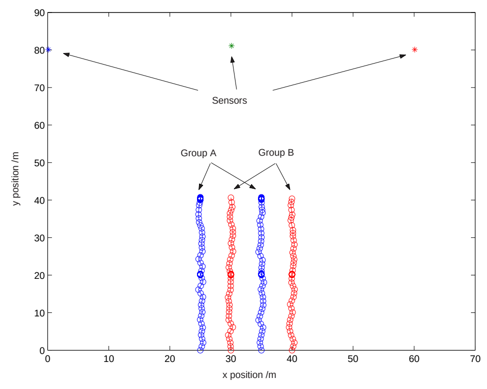

Figure 9: Complete motion of the four targets in our leapfrog scenario together with the positioning of the three angle only sensors.

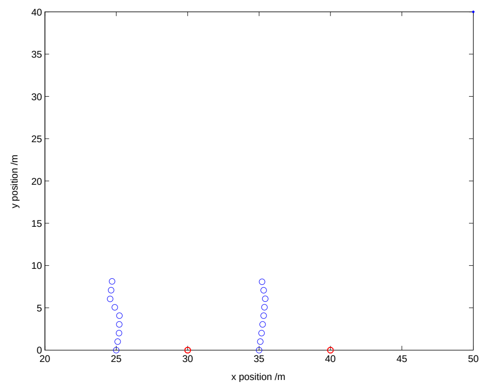

Figure 10: Time steps $k = 1, \ldots, 9$ of the leapfrog motion. Group A moves forward according to kinematic constraints, while group B remains stationary and provides cover.

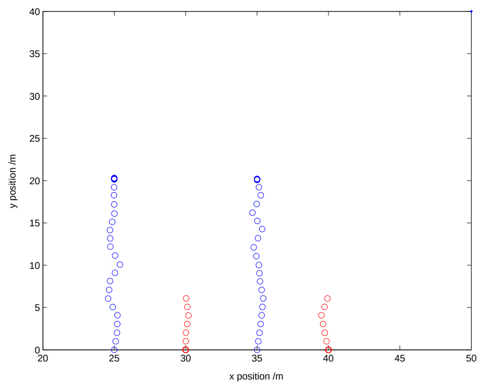

Figure 11: Time steps $k = 1, \ldots, 27$. Group A has reached a certain point and the targets take cover and remain stationary while firing at the enemy. Group B advances forward making use of the cover provided by group A.

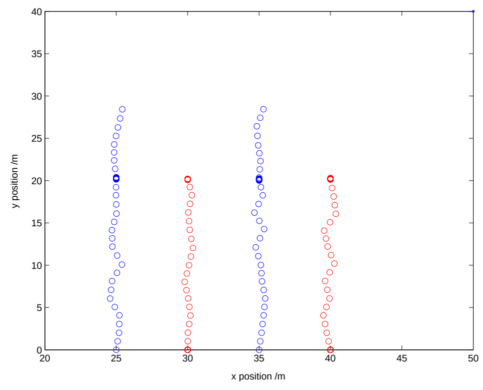

Figure 12: Time steps $k = 1, \ldots, 49$. Group B has stopped moving forward and now provides cover for group A that has started moving forward again.

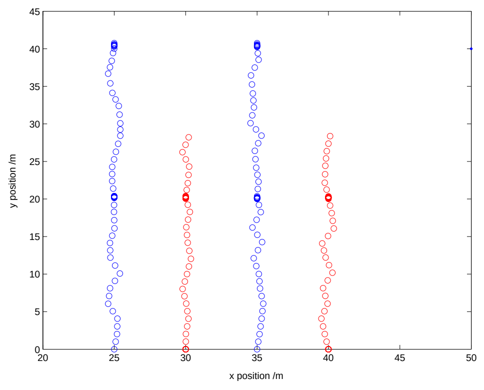

Figure 13: Time steps $k = 1, \ldots, 69$. Group A has reached its final destination and provides cover for group B, which has started moving as soon as Group A started providing cover.

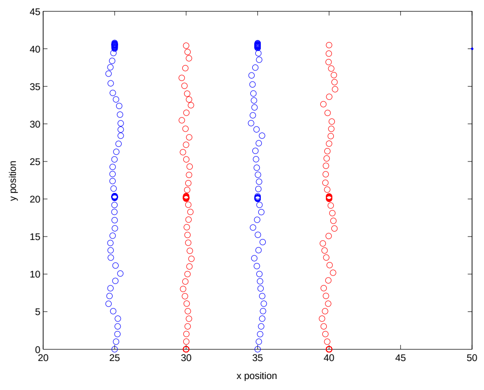

Figure 14: Time steps $k = 1, \ldots, 81$. Group B has also reached its final destination for this scenario.

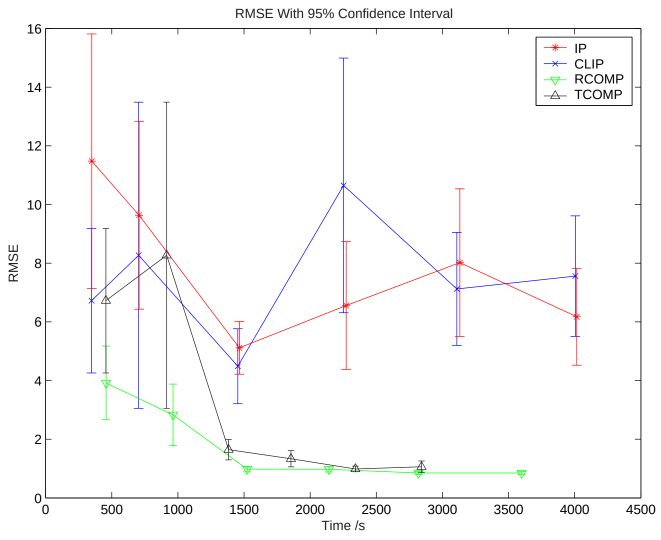

Figure 15: RMSE vs the time that the simulation takes to complete for 81 time steps.

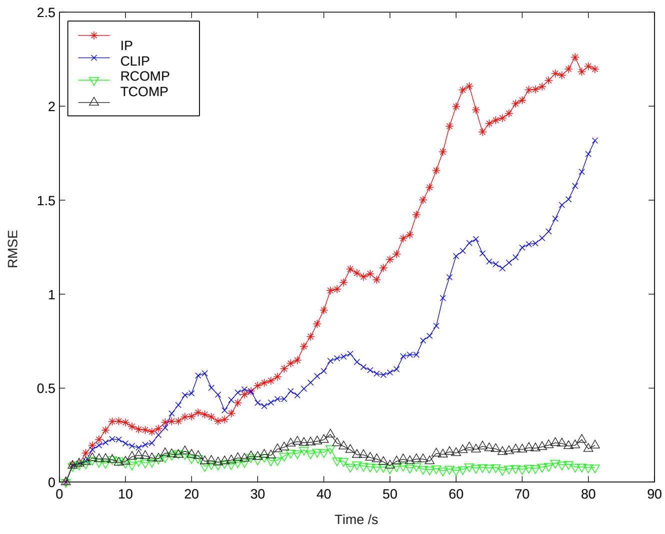

Figure 16: RMSE vs the time step for the entire motion of the targets.

TCOMP. This shows the importance of incorporating kinematic dependency information when available. CLIP still fails to reach the levels of performance of the RCOMP and TCOMP due to its ineffective use of the information provided. As expected, RCOMP and TCOMP have similar performances. Furthermore, the 95% confidence intervals show a significant reduction in the variance of the estimates when using RCOMP and TCOMP.

In Figure 16, the RMSE performance of the IP, CLIP, RCOMP and TCOMP versus the time index corresponding to each time step in the scenario shows that while the RCOMP and TCOMP have an almost constant RMSE throughout the scenario, the IP and CLIP result in an increasing error as the time index progresses. This is attributed to the fact that without the valuable motion constraint information, in the case of the IP, and due to the ineffective use of that information, in the case of the CLIP, as the time index progresses, the trackers loose more tracks and the lost tracks diverge more from the true target positions. As in the convoy scenario, we are comparing the RMSE performance vs the time index taking numbers of particles resulting in almost the same computational expense for all the algorithms.

9 Conclusions and Extensions

9.1 Conclusions

In this report, we proposed the constrained motion proposal (COMP) algorithm that incorporates kinematic constraint information into a particle filter. Kinematic constraint information comes from modeling the possible dependencies in the motion of the targets. Such information exists when targets impose specific types of constraints on the motion of other targets. Therefore, incorporating such information efficiently into the tracker, may reduce the uncertainty in a scenario and improve tracking performance. The COMP uses sampling methods and likelihood functions that take into account motion constraint information. It also introduces variability in partition proposal order. The latter reduces the errors introduced by occasional incorrect information on the constraints imposed on tracked targets. We showed, through simulation results on RMSE performance, that the techniques used by the COMP are effective in incorporating motion constraint information into the tracker. Moreover, that the COMP outperforms the IP, which is not able to utilize such information. It also outperforms the CLIP which makes inefficient use of motion constraint information. The gains in error performance by the COMP come at the expense of algorithmic and computational complexity that are higher than those of the IP and CLIP.

By exploring the different challenges that may arise, and by using particle filtering to handle these challenges, we saw that the particle filter proves its ability to perform well in difficult scenarios utilizing many different kinds of information and adapting to changing situations.

9.2 Extensions

For all the algorithms used to provide simulation results in this report we have assumed that the tracker knows the number of targets in the surveillance area and that this number is constant. The algorithms however, can be easily extended to include an uncertainty in the number of targets being tracked, making the scenarios more realistic. This can be accomplished by assigning probabilities of targets appearing and disappearing from the surveillance area and using these probabilities on each particle. Thus, each particle will have a number of partitions that correspond to a belief on the number of targets present, and this number will be related to the number of partitions it had on the previous step. Hopefully, the particles with the correct number of partitions will receive higher weights in the particle weighing step. For the initialization of the filter we can assume that there is a certain number of targets present and propose particles having anywhere from 1 to that number of partitions according to a uniform probability. This is a successful approach for which simulation results are provided in [7].

Another assumption used by the tracker is the knowledge as to which targets exhibit kinematic dependency. This is usually not a realistic assumption. Even though we can assume that we know the nature of the constraints that may be imposed on the targets, this being attributed to the known nature or the targets (trucks, aeroplanes, etc.), we often do not know if constraints exist in a specific scenario, and if they do, we do not know which of the targets exhibit kinematic dependency. For example, in the convoy case, it is possible for some elements of the convoy to separate from the group. This means that those elements will not impose constraints to other elements in the group, as they did while still in the convoy. Similarly, they will not be bound by the same constraints from targets that are still in the convoy. The same issue arises when we have a variable or unknown number of targets, where we do not know whether the newly arrived targets impose constraints or not. These situations can be tackled by including an uncertainty as to whether the constraints exist, and if so, to which targets they apply. If for a certain particle the belief is that two partitions impose constraints on each other, then motion constraint information will be incorporated in their proposal. The choice on whether to apply constraints can be implemented as follows: For each particle, we can apply a probability that constraints exist among a certain pair of partitions, based on whether the two partitions imposed constraints on each other in the previous time step. For example, if the belief incorporated in a particle in the previous time step was that constraints exist among two partitions then, at the current time step, these constraints are still believed to exist, with a certain probability p, and with a probability $1-p$ they are not. Similarly, if in the previous time step constraints were believed not to exist, then at the current time step, the constraints are still believed not to exist with a probability p, and they are believed to exist with probability $1-p$. Usually probability p, the probability that the state of affairs remains unchanged over a certain time step, is higher than .5. Incorporating such uncertainty into the tracker, creates many different constraint regions applied to different particles. This number of possible regions depends on the number of targets that we have and the certainty we have on the dependency of the motion. In cases of large uncertainty as to whether constraints exist in a scenario, we may use an adaptive algorithm similar to the adaptive partitions algorithm (AP) [7], switching between COMP and the IP or CP according to some measure of the certainty as to whether constraints exist. When kinematic constraints apply, we can expect to have significant improvements in performance, comparable to those presented in this report. Similarly to the case when we have an unknown number of targets, the particles that have the correct belief on the constraints applied will receive higher weights, as they will have been proposed more correctly than the rest of the particles. The introduction of such uncertainty may require an additional number of particles, but will relax some unrealistic assumptions.

References

[1] 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, no. 2, pp. 174-188, 2002.

[2] C. Hue, J. Le Cadre, and P. Pérez, "Sequential Monte Carlo Methods for Multiple Target Tracking and Data Fusion", *IEE Transactions On Signal Processing*, vol.50, no.2, 2002.

[3] C. Hue, J. Le Cadre, and P. Pérez, "A Particle Filter to Track Multiple Objects", *Proceedings of the 2001 IEEE Workshop on Multi-Object Tracking*, pp.61 - 68, 2001.

[4] N. Bergman and A. Doucet, "Markov Chain Monte Carlo Data Association for Target Tracking", *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, vol.2, pp. II705 - II708, 2000.

[5] C. Hue, J. Le Cadre, and P. Pérez, "Tracking Multiple Targets with Particle Filtering using Multiple receivers", Target Tracking: Algorithms and Applications, IEE vol.1, pp.6/1 - 6/4, 2001.

[6] M. Orton and W. Fitzgerald, "A Bayesian Approach to Tracking Multiple Targets Using Sensor Arrays and Particle Filters," *IEEE Transactions on Signal Processing*, vol.50, no.2, pp.216-223, 2002.

[7] C. Kreucher, K. Kastella and A. O. Hero III, "Multitarget Tracking Using a Particle Filter Representation of the Joint Multitarget Density", (to be submitted).

[8] L. D. Stone, C. A. Barlow, and T. L. Corwin, *Bayesian Multiple Target Tracking*, Artech House, 1999.

[9] J. MacCormick and A. Blake, "A Probabilistic Exclusion Principle for Tracking Multiple Objects", *Proceedings of the International Conference of Computer Vision*, 1999, pp.572-578.

[10] M. Isard and J. MacCormic, "BraMBLe: A Bayesian Multiple-Blob Tracker", *Proceedings of the 8th International Conference on Computer Vision*, 2001.

[11] I. Kyriakides, D. Morrel and A. Papandreou-Suppappola, "Multiple Target Tracking With Constrained Motion Using Particle Filtering Methods", *Asilomar Conference on Signals, Systems, and Computers*, (to be submitted).

[12] M. Montemerlo, S. Thrun, and W. Whittacker, "Conditional Particle Filter for Simultaneous Mobile Robot Localization and People Tracking", *Proceedings of the IEEE Conference on Robotics and Automation*, vol.1, pp.695-701, 2002.

[13] D. Tweed and A. Calway, "Tracking Multiple Animals in Wildlife Footage", *Proceedings of the Conference on Pattern Recognition*, vol.2, pp. 24-27, 2002.

[14] D. Schulz, W. Burgard, D. Fox and A. B. Cremers, "Tracking Multiple Moving Targets with a Mobile Robot using Particle Filters and Statistical Data Association", *Proceedings of the IEEE International Conference on Robotics and Automation*, 2001.

[15] Citation styles [online], "Convoy Operations Manual", 2001, `http://www.bedfordstmartins.com/online/citex.html` (Accessed: 2005).

[16] Citation styles [online], "Air Assault Division Operations", 1996, `http://www.bedfordstmartins.com/online/citex.html` (Accessed: 2005).

[17] S.S. Blackman, "Multiple Target Tracking With Radar Applications", Archtech House, Norwood, MA, 1986.

[18] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems", *Transactions of the ASME–Journal of Basic Engineering,* vol. 82, series D, pp.35-45, 1960.

[19] C. R. Rao, C. R. Sastry and Y. Zhou, "Tracking the Direction-of-Arrival of Multiple Moving Targets", *IEEE Transactions on Signal Processing*, vol. 42, Issue 5, pp.1133 - 1144, 1994.

[20] A. H. Jazwinsky, "Stochastic Processes and Filtering Theory", New York: Academic Press, 1970.

[21] A. J. Julier, and J. K. Uhlman, "A New Extension of the Kalman Filter to Nonlinear Systems", *Proceedings of Aerosence: The Eleventh International Symposium on Aerospace/Defence Sensing, Simulation and Controls*, vol. 3068, pp.182-193, 1997.

[22] A. Doucet, S. Godsill, and C. Andreu, "On Sequential Monte Carlo Sampling Methods for Bayesian Filtering", *Statistics and Computing*, vol.10, no.3, pp.197-208, 2000.

[23] A. Doucet, J. F. G. de Freitas, and N. J. Gordon, "An Introduction to Sequential Monte Carlo Methods", *Sequential Monte Carlo Methods in Practice*, A. Doucet, J. F. G. de Freitas, and N. J. Gordon, Eds. New York: Springer-Verlag, 2001.

[24] J. Carpender, P. Clifford, and P. Fearnhead, "Improved Particle Filter for Nonlinear Problems", *Proceedings of the Institute of Electrical Engineering, Radar, Sonar, Navigation*, 1999.

[25] N. Gordon, D. Salmond, and A. F. M. Smith, "Novel Approach to Nonlinear, and Non-Gaussian Gayesian State Estimation", *Proceedings of the Institute of Electrical Engineering*, vol.140, pp.107-113, 1993.

[26] D. Crisan, P. Del Moral, and T. J. Lyons, "Nonlinear Filtering Using Branching and Interacting Particle Systems", *Markov Processes Related Fields*, vol.5, no.3, pp.293-319, 1999.

[27] P. Del Moral, "Nonlinear Filtering: Interacting Particle Solution", *Markov Processes Realted Fields*, vol.2, no.4, pp.555-580.

[28] K. Kanazawa, D. Koller, and S. J. Russell, "Stochastic Simulation Algorithms for Dynamic Probabilistic Networks", *Proceedings of the Eleventh Annual Conference of Uncertainty AI*, 1995, pp.346-351.

[29] N. Bergman, "Recursive Bayesian Estimation: Navigation and Tracking Applications", Ph.D. dissertation, Linköping Univ., Linköping, Sweden, 1999.

[30] D. Crisan and A. Doucet, "A Survey of Theoretical Results on Particle Filtering for Practitioners", *IEEE Trans. Signal Processing, to appear*, 2002.

[31] G. Kitawa, "Monte Carlo Filter and Smoother for Non-Gaussian Nonlinear State Space Models", *Journal of Computational and Graphical Statistics*, pp.1-25, vol. 5(1), 1996.

[32] J. S. Liu and R. Chen, "Sequential Monte Carlo Methods for Dynamical Systems," *Journal of American Statistical Association*, vol. 93, pp.1032-1044, 1998.

[33] T. E. Fortman, Y. Bar-Shalom, and M. Scheffe, "Sonar Tracking of Multiple Targets Using Joint Probabilistic Data Assciation", *IEEE J. Oceanic Engineering*, vol.OE-8, pp.173-184, 1983.

[34] J. Y. He, Y. Yan, Q. Meng, "Multiple Arrays Multiple Targets Data Fusion and Tracking Based on TDOA Measurement Using FCM-JPDA Algorithm", *Proceedings of the 2002 ICSP*, vol.6, pp. 7803-7488, 2002.

[35] J. K. Tugnait, "Tracking of Multiple Maneuveiring Targets in Clutter Using Multiple Sensors, IMM and JPDA Coupled Filtering", *Proceedings of the American Control Conference*, 2003.

[36] D. B. Reid, "An Algorithm for Tracking Multiple Targets", *IEEE Transactions on Automatic Control*, vol.AC-24, pp.843-854, 1979.

[37] M. L. Krieg and D. A. Gray, "Multi-Sensor, Probabilistic Multi-Hypothesis Tracking", *First Australian Data Fusion Symposioum*, pp. 153 - 158, 1996.

[38] E. Giannopoulos and R. Streit, "Probabilistic Multi-Hypothesis Tracking in a Multi-Sensor, Multi-Target Environment", *Data Fusion Symposium*, pp. 184-189, 1996.

[39] C. Rago, P. Willett and R. Streit, "Direct data fusion using the PMHT", *Proceedings of the American Control Conference*, vol.3, pp. 1698 - 1702, 1995.

[40] K.M. Alexiev, "Multiple Target Tracking Using Hough Transform PMHT Algorithm", *Proceedings of the First International IEEE Symposium on Intelligent Systems*, vol.1, pp.227 - 232, 2002.

[41] Y. Bar-Shalom, *Multitarget-Multisensor Tracking: Advanced Applications*, Artech House, 1990.

[42] Y. Bar-Shalom and T. E. Fortmann, *Tracking and Data Association*, Artech House, 1988.

[43] H. A. P. Blom and E. A. Bloem, "Joint IMMPDA Particle Filter", *Proceedings of the 6th International Conference on Information Fusion*, Cairns, Queensland, Australia, 2003.

[44] R. Karlsson and F. Gustafsson, "Monte Carlo Data Association for Multiple Target Tracking", *Proceedings of the IEE Workshop on Target Tracking: Algorithms and Applications*, The Netherlands, 2001.

[45] K. Kastella, "Joint Multitarget Probabilities for Detection and Tracking", *Proceedings of SPIE Conference in Acquisition, Tracking, and Pointing XI*, Orlando, 1997.

[46] S. Maskel, M. Rollason, N. Gordon, D. Salmond, "Efficient Particle Filtering for Multiple Target Tracking with Application to Tracking in Structured Images", *Proceedings of SPIE Conference on Signal and Data Processing of Small Targets*, Orlando, 2002.

[47] C. Kreucher, K. Kastella and A. O. Hero III, "Tracking Multiple Targets Using a Particle Filter Representation of the Joint Multitarget Probability Density", *Proceedings of SPIE International Symposium on Optical Science and Technology*, Orlando, 2003.

[48] N.J. Gordon, D.J. Salmond and D. Fisher, "Bayesian Target Tracking After Group Pattern Distorition", *Signal and Data Processing of Small Targets*, SPIE vol. 3163, pp.238-248, 1997.

[49] D.J. Salmond and N.J. Gordon, "Group and Extended Object Tracking", *Signal and Data Processing of Small Targets*, SPIE vol.3809, 1999.

[50] N. Ikoma and S. Godsill, "Extended Object Tracking With Unknown Association, Missing Observations, and Clutter, Using Particle Filters", *2003 IEEE Workshop on Statistical Signal Processing*, pp.502-505, 2003

[51] J. Vermaak, N. Ikoma and S. Godsill, "Extended Object Tracking Using Particle Techniques", *2003 IEEE Proceedings Aerospace Conference*, vol.3, pp.1885, 2004.

[52] A. Papoulis and S.U. Pillai, "Probability, Random Variables and Stochastic Processes", *McGraw-Hill Education*, 2001