Robust Tracking of Soccer Robots Using Random Finite Sets

Pablo Cano and Javier Ruiz-del-Solar, *AMTC Center, Universidad de Chile*

Maintaining a good estimation of the other robots' positions is crucial in soccer robotics. To tackle the data association problem, the proposed approach doesn't require explicit data association and integrates information shared by teammate robots.

Multirobot systems correspond to groups of intelligent robots that perceive and act in a given environment to collectively solve a task. The robots can use different cooperation and coordination mechanisms, which can include cooperative perception, distributed world modeling, planning and control, and distributed decision making, among others. There are many application areas of multirobot systems such as surveillance, exploration, warehouse management, transportation, rescue applications, and soccer robotics, just to name a few.

Soccer robotics corresponds to a very well-known benchmark of multirobot systems, in which two teams of autonomous robots play soccer against each other. In the most advanced soccer leagues (Standard Platform League [SPL] and Humanoid League; www.robocup.org/robocup-soccer), each robot player perceives the environment individually, shares its observations with its teammates, models the environment, which includes the modeling of dynamic objects such as the ball and the other robot players, makes decisions in a centralized or distributed fashion, and acts (that is, it plays soccer). The focus of our research is on soccer robotics, specifically, the robust tracking of robot players by a robot. This is an interesting tracking problem in which both teammates and opponents need to be tracked. Teammates normally share their positions and their estimations of the other robots' positions, but opponents don't share any information.

When playing soccer, one of the main challenges is maintaining a good estimation of opponent and teammate positions to select appropriate behaviors. High-level behaviors such as passing, team formation, and role decisions, just to name a few, need information about players' positions on the field. This problem is particularly complex in leagues that use humanoid robots such as the SPL and the Humanoid League: the number of robots in the field changes constantly (robots can be removed from the field due to penalizations or failures), robots of the same team are indistinguishable, the small field of view of the cameras allows only a restricted or partial observation of the field, and the robots have limited computational capabilities. These limitations can be overcome by sharing observations among teammates and by a robust estimation of the robots' positions. The estimation of those positions, which is denoted as robot tracking, has been tackled in a variety of ways within the robot soccer community. Bayesian filtering approaches such as multiple hypothesis tracking (MHT),[1] extended Kalman filters (EKFs),[2] the multihypothesis unscented Kalman filter,[3] and Gaussian mixture models[4] have been proposed to address this problem. However, all these approaches involve an explicit association between measurements and targets, using heuristics or statistics. This step is commonly known as the data association problem.

In this context, the main goal of this article is to propose a new methodology for the robust tracking (position estimation) of multiple soccer robots using the Random Finite Sets (RFS) framework, which doesn't require explicit data association. The proposed methodology is inspired by another work[5] where the term probability hypothesis density (PHD) was introduced as the first moment of a point process. The PHD filter is proposed as a way to maintain hypotheses of multiple objects using finite sets instead of vectors to describe object states.[6] In fact, this paradigm considers the unknown number of objects to be tracked (robots, in our case) as a multi-object set represented by an object and the measurements received by the sensor as a single set of observations, which it models as RFS. RFS is a set of random variables, for which the cardinality is itself a random variable.

The RFS framework has been used to track features in SLAM (Simultaneous Localization and Mapping) applications as well as to track multiple objects and targets.[7,8] However, this is the first application of RFS in soccer robotics, and one of the first that addresses the tracking of robots in a dynamic environment, along with other works[9,10] that track moving objects. The main contributions here are the application of the RFS framework to the tracking of multiple robots in a highly dynamic environment and the incorporation in the tracking process of information shared by teammates' robots.

The proposed robot tracking methodology is validated in simulated robot soccer games, using simulated Nao robots, and compared against a classical, state-of-the-art multihypothesis EKF tracking methodology.[3] The specific application of the proposed tracking methodology is soccer robotics, but it can be adapted to track robots in other applications and environments.

Problem Description: Data Association when Tracking Multiple Players in Robot Soccer

As already mentioned, knowing the position of the other robots in the field is relevant for implementing high-level soccer behaviors. In this work, we will use *local map of robots* for a map that a given player builds and that includes the positions on the field of every other robot player (teammate or opponent). We denote observations of robot detections and hypotheses for the estimated positions of the robots on that map.

Most existing methods for estimating the local map of robots employ a classical approach with a vector representation of the robot hypotheses, which are propagated over time using a Bayesian filter (such as an EKF filter). However, it has been demonstrated that the use of a vector representation of hypotheses has numerous drawbacks mainly related to the data association between new and past observations (hypotheses).[11]

Multitarget Tracking with RFS

The main idea of the proposed methodology is to use finite sets instead of vectors to represent both observations and hypotheses. As has been widely demonstrated,[6,11] the first moment of RFS, known as PHD, can be used to construct a filter that propagates PHD to the map posterior instead of the map posterior itself.

PHD Filter

The PHD function $v(m)$ at position m represents the density of the expected number of objects (robots) occurring at that position of the state space (map). Therefore, a property of the PHD is that for any given region S of the map:

$$\mathbb{E}\left[|M \cap S|\right] = \int_S v(m)\,dm, \tag{1}$$

where M represents the RFS map and $|\cdot|$ denotes a set's cardinality. This means that, by integrating the PHD on any region S of the map, we obtain the expected number of objects in S.[6]

The PHD filter considers the following two steps:[6]

- Prediction:

$$v_{k|k-1}(m) = v_{k-1}(m)\, b_k(m), \tag{2}$$

where $v_{k-1}(m)$ is the previous estimate of the PHD, $v_{k-1}(m)$ is its prediction at time k, and $b_k(m)$ represents the PHD of the new objects at time k. This equation is a simplification of the prediction step shown elsewhere,[6,12] where the targets are supposed to be stationary and don't spawn new targets. This is done because the omnidirectional movement of the robots is very difficult to model from the observational point of view. That is why a zero-order modeling is used (zero mean plus covariance).

- Update:

$$v_k(m) = v_{k|k-1}(m)\left[1 - P_D(m)\right] + v_{k|k-1}(m) \left[\sum_{z \in Z_k} \frac{P_D(m)\, g_k(z \mid m))}{c_k(z) + \int P_D(\xi)\, g_k(z|\xi)\, v_{k|k-1}(\xi)\, d\xi}\right], \qquad (3)$$

where z represents a measurement, Z_k is the set of all measurements obtained at time k, $P_D(m)$ represents the probability of detecting z at m, $g_k(z|m)$ represents the likelihood that z is generated by an object at m (that is, the observation likelihood), and $c_k(z)$ is the PHD of the clutter intensity.

Thus, when computing the posterior PHD $v_k(m)$, the first term corresponds to the predicted objects weighted by their probabilities of missed detection and the second to the predicted objects, updated by the spatial locations of all the new observations, and their probabilities of detection.

Using Multiple Sensors

The filter mentioned above works for single-sensor scenarios, but for two or more sensors, the PHD corrector equation become computationally intractable.[13] Given this limitation, Ronald Mahler proposed an *iterated-corrector approximation*:[13] during each measurement cycle, iterate the PHD filter equations once for each sensor. This approximation has been proved to be stable and doesn't result in noticeable differences in performance.[13] So, with these changes, the PHD filter equations are

- Prediction:

$$v_{k|k-1}(m) = v_{k-1}(m) + b_k^{(i)}(m). \qquad (4)$$

- Update:

$$v_k(m) = v_{k|k-1}(m) \left[1 - P_D^i(m) + \sum_{z \in Z_k^{(i)}} \frac{P_D^{(i)}(m)\, g_k^{(i)}(z \mid m))}{c_k^{(i)}(z) + \int P_D^i(\xi)\, g_k^i(z|\xi)\, v_{k|k-1}(\xi)\, d\xi}\right], \qquad (5)$$

where i is the sensor index, and $Z_k^{(i)}$ are the observations obtained by the ith sensor.

To adapt this framework to the multiple object tracking by a mobile robot problem, some considerations must be made. As presented earlier, $P_D^{(i)}(m)$ represents the probability of detecting object at m by the ith sensor, but it doesn't consider its dependence on the current robot position. So, the real probability is represented by $P_D^{(i)}(m|X_k)$, where X_k represents the position of the robot at time k. The same dependence on X_k applies to $g_k^{(i)}$, $c_k^{(i)}$, and $b_k^{(i)}$ because they also depend on robot position.

Tracking Robots Using RFS

The proposed robot tracking system is based on the PHD filter, and it considers the robots' observations of their own cameras and robot poses received from teammate robots to build the local map of robots (see Figure 1). In addition, the system can merge local maps received from teammate robots to build a combined map of robots.

The tracking system uses a *mixture of Gaussians* implementation of the PHD filter (GM-PHD)[12] because it's time efficient and able to run in real time in robots with limited computational resources.

GM-PHD Implementation

We want to represent the likelihood of any RFS, that is, the PHD, as a mixture of Gaussians, so both hypotheses and new observations are represented by Gaussians. But to allow Gaussians to represent the position and number of objects (robots) in the field, it's necessary to add a weight to every Gaussian. In this way, the Gaussian mean values represent the different locations of the hypotheses on the map, while their weights represent the number of hypotheses in a given region. So, a PHD map is a Gaussian mixture of the form

$$v_{k-1}(m|X_{k-1}) = \sum_{j=1}^{J_{k-1}} \omega_{k|k-1}^{(j)} \mathcal{N}\left(m;\, \mu_{k|k-1}^{(j)},\, P_{k|k-1}^{(j)}\right), \qquad (6)$$

which is a mixture of J_{k-1} Gaussians, with $\omega_{k|k-1}^{(j)}$, $\mu_{k|k-1}^{(j)}$, and $P_{k|k-1}^{(j)}$ being the weight, mean, and covariance, respectively. The same form is used to represent the new hypotheses at time k, $b_k^{(i)}(m \mid X_k)$:

$$b_k^{(i)}(m|X_k) = \sum_{j-1}^{J_{b,k}} \omega_{b,k}^{(j)} \mathcal{N}\left(m;\, \mu_{b,k}^{(j)},\, P_{b,k}^{(j)}\right), \qquad (7)$$

where $J_{b,k}$ is the number of Gaussians in the new PHD at time k, and $\omega_{b,k}^{(j)}$, $\mu_{b,k}^{(j)}$, and $P_{b,k}^{(j)}$ are the weight, mean, and

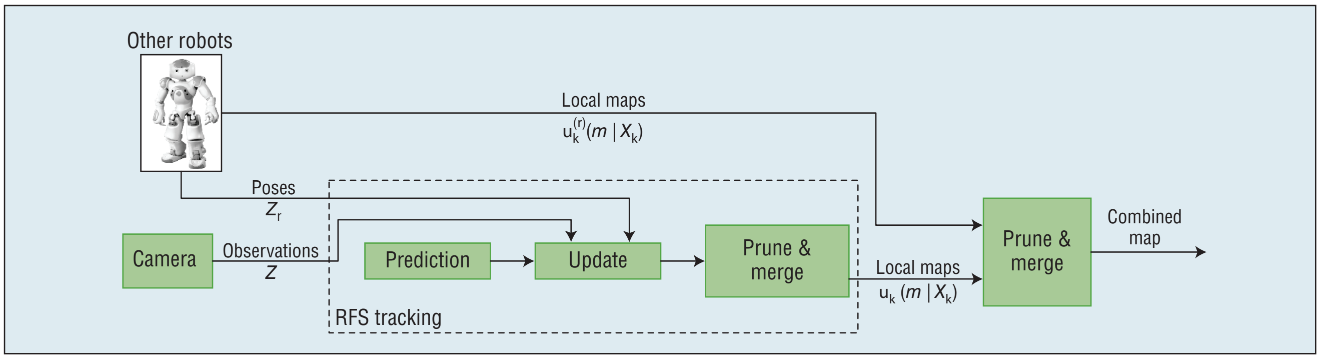

Figure 1. General representation of the proposed method.

covariance, respectively. The calculation of this PHD is made by using the previous observations of each sensor, where each observation creates a Gaussian of the GMM, implying that $J_{b,k} = |Z_{k-1}|$. Therefore, the predicted PHD of the map is also a Gaussian mixture given by

$$v_{k|k-1}(m|X_k) = \sum_{j=1}^{J_{k|k-1}} \omega_{k|k-1}^{(j)} \mathcal{N}\left(m;\ \mu_{k|k-1}^{(j)}, P_{k|k-1}^{(j)}\right), \tag{8}$$

where $J_{k|k-1} = J_{k-1} + J_{b,k}$ is the number of Gaussians representing the union of the prior PHD map $v_{k-1}(m|X_{k-1})$, and the new PHD hypotheses at time k. Note that $\omega_{k|k-1}^{(j)}$, $\mu_{k|k-1}^{(j)}$, and $P_{k|k-1}^{(j)}$ are the weight, mean, and covariance of the Gaussians of the prior map PHD if $j \le J_{k-1}$, and the weight, mean, and covariance of the Gaussians of the new observations PHD otherwise.

So, the posterior PHD given by Equation 5 is also a Gaussian mixture of the form

$$v_k(m|X_k) = v_{k|k-1}(m|X_k)\left[1 - P_D^{(i)}(m|X_k)\right] + \sum_{z \in Z_k} \sum_{j=1}^{J_{k|k-1}} v_{G,k}^{(i,j)}(z,\ m|X_k) \quad , \tag{9}$$

where, according to the general PHD filter update equation, $v_{G,k}^{(i,j)}$ corresponds to

$$v_{G,k}^{(i,j)}(z,\ m|X_k) = \omega_k^{(i,j)}(z \mid X_k)\mathcal{N}\left(m;\ \mu_{k|k}^{(j)},\ P_{k|k}^{(j)}\right) \tag{10}$$

with

$$\omega_k^{(i,j)}(z|X_k) = \frac{P_D^{(i)}(m|X_k)\,\omega_{k|k-1}^{(j)}\,q^{(i,j)}(z \mid X_k)}{c_k(z) + \sum_{l=1}^{J_{k|k-1}} P_D^{(i)}(m|X_k)\,\omega_{k|k-1}^{(l)}\,q^{(i,l)}(z \mid X_k)}, \tag{11}$$

where $q^{(i,j)}(z \mid X_k) = \mathcal{N}\left(z;\ H_k^{(i)}\mu_{k|k-1}^{(j)},\ S_k\right)$ is the observation likelihood, i is the sensor index, and j the Gaussian index.

The components $\mu_{k|k}^{(j)}$ and $P_{k|k}^{(j)}$ can be obtained from the standard EKF update equations:

$$S_k^{(l)} = R_k^{(i)} + \nabla_x H_k^{(i)} P_{k|k}^{(l)} \left(\nabla_x H_k^{(i)}\right)^T \tag{12}$$

$$K_k^{(l)} = P_{k|k}^{(i)} \left(\nabla_x H_k^{(i)}\right)^T \left[S_k^{(l)}\right]^{-1} \tag{13}$$

$$\mu_{k|k}^{(l)} = \mu_{k|k-1}^{(l)} + K_k^{(l)}\left(z - H_k^{(i)}\left(\mu_{k|k-1}^{(l)}\right)\right) \tag{14}$$

$$P_{k|k}^{(l)} = \left[I - K_k^{(l)} \nabla_x H_k^{(i)}\right] P_{k|k-1}^{(l)}, \tag{15}$$

with $\nabla_x H_k^{(i)}$ representing the Jacobian of the observation model with respect to the estimated location for the ith sensor.

The parameters used for the creation of the $b_k^{(i)}(m|Z_{k-1}, X_k)$ Gaussians are

$$\omega_{b,k}^{(j)} = \theta_{weight}, \qquad \mu_{b,k}^{(j)} = h_i^{-1}\left(z_{k-1}^j, X_k\right), \\ P_{b,k}^{(j)} = \nabla_m^j H_k^{(i)}\ R^{(i)} \left[\nabla_m^j H_k^{(i)}\right]^T \tag{16}$$

where h_i^{-1} is the inverse observation model of the ith sensor, $R^{(i)}$ is the observation noise covariance of the ith sensor, $\nabla_m^j H_k^{(i)}$ is the Jacobian of the observation model of the ith sensor, with respect to the Gaussian state j, and θ_{weight} is the initial weight.

Then, as every Gaussian hypothesis is combined with every observation to generate new Gaussian hypotheses,

```
//the parameters of the Map's MoG model $(v_{k-1}(m|X_{k-1}))$ are modified
for $i = 1$ to $J_{k-1}$ do
  //the robots may move -> covariance is increased using the movement noise
covariance Q
  $\mu^{(i)}_{k|k-1} = \mu^{(i)}_{k-1}, P^{(i)}_{k|k-1} = P^{(i)}_{k-1} + Q, w^{(i)}_{k|k-1} = w^{(i)}_{k-1}$
end for
//birth; observations are added; robots' position received from other robots are
treated as observations from additional sensors
for each sensor $i$
  generateNewGausians$(Z_{k-1}, X_{k-1})$ //use eq.(7) and (16)
  $v_{k|k-1}(m|X_k) = \{\mu^{(j)}_{k|k-1}, P^{(j)}_{k|k-1}, w^{(j)}_{k|k-1}\}^{J_{k|k-1}}_{j=1}$
  //update step
  for $j = 1$ to $J_{k|k-1}$ do
    calculate $P_D^{(i,j)}$
    $w^{(j)}_k = (1 - P_D^{(i,j)})\, w^{(j)}_{k|k-1}$
  endfor
  $N = 0$
  for each $z$ in $Z_k$
  for $j = 1$ to $J_{k|k-1}$ do
    calculate $H^i, S^{(j)}_k$ and $K^{(j)}_k$
    $\mu^{(N+j)}_k = \mu^{(j)}_{k|k-1} + K^{(j)}_k(z - \mu^{(j)}_{k|k-1})$
    $P^{(N+j)}_k = [I - K^{(j)}_k H^i]P^{(j)}_{k|k-1}$
    $\tau^{(j)} = P_D^{(i,j)} w^{(i)}_{k|k-1} |2\pi S^{(i)}_k|^{-0.5}$
                          $\times \exp\left((z - \mu^{(j)}_{k|k-1})[S^{(j)}_k]^{-1}(z - \mu^{(j)}_{k|k-1})^T\right)$
  end for
  for $j = 1$ to $J_{k|k-1}$ do
    $w^{(N+j)}_k = t^{(j)} / \left(c(z) + \sum_{l=1}^{J_{k|k-1}} t^{(l)}\right)$
  end for
  $N = N + J_{k|k-1}$
  end for
  $J_k = N$
  //updated map
  $v_k(m|X_k) = \{\mu^{(j)}_k, P^{(j)}_k, w^{(j)}_k\}^{J_k}_{j=1}$
  merge_and_prune $(v_k(m|X_k))$ // use eq. (18)
end for
//combined map building; local maps received from other robots are merged with
the local map
for each received local map $i$
  merge_maps $(v_k(m|X_k), v^{(i)}_k(m|X_k))$ //use eq. (19)
end for
```

Figure 2. Pseudocode of the general algorithm that calculates the PHD that represent the map of robots.

the numbers of Gaussians may grow exponentially in every frame. That is why pruning and merging operations are necessary.[13] First, all Gaussians are sorted by weight. Then, for each Gaussian g_i from $v_k(m|X_k)$,

$$\text{merge}(g_i, g_j) \text{ if } dM(g_i, g_j) < \theta_{local}, \forall g_j \in v_k, j > i, \tag{17}$$

where $dM(g_i, g_j)$ is the Mahalanobis distance between g_i and g_j, and θ_{local} is the local distance threshold. Note that this merge doesn't represent an elimination of a hypothesis because one Gaussian can represent more than one hypothesis by its weight, which are added when two or more Gaussians are merged. The Mahalanobis distance is used because it considers the covariance of the Gaussians being compared. Also, if a Gaussian has a weight below a given threshold $\theta_{minWeight}$, then it's erased from the map.

This process is called merge_and_prune in Figure 2.

Construction of the Local Map of Robots

Multiple aspects are needed to construct the local map of robots using the presented framework (GM-PHD).

Observation process. The robot that builds the map has two sources of information: its own camera and the information sent by its teammates. To detect the other robots in the input images, color blobs are built using scan lines. Blobs surrounded by green pixels are robot candidates (because the field is always green as a rule). Additional details of this detector can be found elsewhere.[3] Then, if one or more robots are detected in the image, their positions are projected on the field coordinates by using the camera's forward kinematic, its intrinsic and extrinsic parameters, and the robot's pose. Also, depending on the position of the detected robots relative to the observing robot, a covariance is calculated, which is also projected on the field coordinates.[3] This covariance is used as $R^{(1)}$ (the covariance of the first sensor). The probability of detection p_D and the clutter intensity are calculated empirically, using statistics of the false and true positives in several situations. The p_D of each Gaussian of the map doesn't have a constant value and must be recalculated in every frame by using the inverse kinematics of the camera, that is, the information of where the camera is observing. If the robot is expected to appear in the image, then it has the p_D calculated before. But if not, the p_D is set to zero.

The second source of information corresponds to the poses of all the teammates, each one treated as a simulated sensor. This information is available through wireless communication. Each robot shares not only its pose but also its player number (every robot on a team must have a number between 1 and 7). So, to use this information in the GM-PHD framework, two other variables are added to each Gaussian. The first is a Boolean (called g^c) that indicates if this Gaussian has ever been updated using the communication information (that is, the other robot poses), and the second is the player number that was used to update this Gaussian (called g^p). This second variable is considered valid only when $g^c == true$. So, when the pose of a teammate is added to the map as defined in Equation 7, this Gaussian starts with $g^c == true$ and g^p equal to the number of the sharing teammate. For these cases, the merge operation defined by Equation 17 is modified as

$$\begin{aligned}&\text{merge}\left(g_i, g_j\right) \quad \text{if } [\{\sim\left(g_i^c \text{ and } g_j^c\right) \text{ and } \text{dM}\left(g_i, g_j\right) \\ &< \theta_{\text{local}}\} \text{ or } \left\{g_i^c \text{ and } g_j^c \text{ and } g_i^p \quad = g_j^p\right\}]\end{aligned} \quad . \qquad (18)$$

This implies that Gaussians made from poses of two different teammates are never merged, but Gaussians made from the camera sensor can be merged with any other Gaussian that's close enough. This also lets us calculate a detection probability that depends on whether the Gaussian was communicated one or not, that is, the p_D is set on zero for this simulated sensor if $g^c == \mathit{false}$ and near 1 if $g^c == \mathit{true}$.

So, when the final local map is constructed, each target can be associated with one of the teams (their own team or the rival team) by using the Boolean g^c. This information is critical if the local map is going to be used for a high-level task such as passing, and it can't be accomplished without information about the teammates' positions.

Modeling. The state space used in this approach is a vector $p = (x, y)$ that indicates the robot's position in the field coordinates, whose origin is the center of the field. To use this state space, it's necessary to assume that the robot is always localized. In the RoboCup competition, this is generally true because this is the most important subject for a robot to play properly. Particularly in this case, the robots use a multihypothesis Kalman filter for self-localizing.[14,15]

Given that both sensors already give their observations in the field coordinates, H is the identity matrix for both sensors. Also, the use of this state space implies that the speed of the robots isn't used as part of the state. This is the reason for using the simplified version showed in Equation 2.

Construction of the Combined Map of Robots

To build a combined map of robots for each robot, the robots must share their local maps, that is, the GM representation of the map. Thus, the local maps are combined by using the weight and positions of every Gaussian using a pruning and merging process.[12] This process orders every Gaussian by its weight and then uses the Mahalanobis distance to merge similar hypotheses into one. So, for each Gaussian g of every local map $v_k^{(i)}(m|X_k)$ of each other robot i,

$$CMap \leftarrow \begin{cases} g, & dM\left(g, g_{j\in LMap}\right) > \theta_{combined} \\ merge\left(g, g_j\right), & dM\left(g, g_{j\in LMap}\right) < \theta_{combined} \end{cases}, \qquad (19)$$

where *CMap* is the combined map, *LMap* is the local map of the current robot, that is, $v_k(m|X_k)$, $\text{dM}(g_i|g_j)$ is the Mahalanobis distance between g_i and g_j, and $\theta_{combined}$ is the distance threshold. This process is called merge_maps in Figure 2.

Results

We evaluated the proposed tracking methodology in simulated soccer matches of Nao robots (1 versus 1 and

Table 1. Summary of the obtained results. The numbers correspond to the OSPA values of each experiment. All values are in millimeters (mm).

	Test	Average	Best	Worst
RFS-Local-Map	1 vs. 1	197.8	182.4	212.4
	5 vs. 5	348.6	319.7	381.9
RFS-Combined Map	5 vs. 5	271.3	225.2	298.7
MH-EKF	1 vs. 1	293.6	271.1	305.3
	5 vs. 5	421.5	391.3	453.2

5 versus 5). The main reason for using simulations is to repeat experiments and get an accurate measurement of the ground truth of the robots' positions. Nevertheless, it's important to mention that the proposed tracking system is used in our soccer team of Nao robots, where it can run in real time (approximately 30 fps).

The Nao robot has two cameras of 60° horizontal field of view and 47° vertical field of view, and uses a single-core ATOM processor running at 1.6 GHz. The B-Human Simulator,[16] a 3D tool that accurately simulates the physics and sensors in the Nao robot, is used in the experiments. Note that this tool simulates the images that the robot receives, therefore perception procedures are made using these images exactly as in a real robot, meaning that the simulation has a very similar noise process of the detected robots.

Two variants of the proposed method are analyzed, RFS-Local-Map, which doesn't consider maps received from other robots, and RFS-Combined-Map, which corresponds to the proposed method. The methods are tested against a classical multihypothesis EKF (MH-EKF) tracking method,[3] where each robot position is estimated via an independent EKF. In the method, each observation is associated with an existing hypothesis using a distance threshold D_{EKF}; a new hypothesis is created if the observation isn't associated. If a hypothesis has no observations in a certain period T_{EKF}, then it's eliminated.

To evaluate the performance of each method, we used the OSPA metric.[17] This metric allows us to compare the obtained maps of robots with the ground truth map of robots by calculating a distance $\bar{d}^c$ between the two sets (maps) as

$$\bar{d}^c(X,Y) = \left(\frac{1}{|Y|} \left(\min_{j \in \{1,..,|Y|\}} \sum_i^{|X|} d^c\left(x^i, y^j\right)^2 + c^2\left(|Y| - |X|\right) \right) \right)^{1/2}, \quad (20)$$

where $|X| < |Y|$, $d^c(x^i, y^j) = \min(c, \|x^i - y^j\|)$, and c is a cut-off parameter that represents the maximum distance to associate two positions.

The first experiments correspond to 10 simulated soccer matches of 5 versus 5 robots. In the experiments, all the robots calculate the OSPA metric for each method under evaluation, with a cut-off value $c = 500$ mm. The MH-EKF parameters used are $D_{EKF} = 500$ mm and $T_{EKF} = 8s$, which are selected to have the best OPSA results. In the case of the RFS-Local-Map, the parameters are $\theta_{weight} = 0.01$, $\theta_{local} = 18$, and $\theta_{object} = 0.3$. For the sensors, $P_D^{(1)} = 0.35$ if the robot must appear in the image and zero if not, and $P_D^{(2)} = 0.98$ if the Gaussian has $g^c = true$ and zero if $g^c = false$. In addition, $R^{(1)}$ is calculated for the perceptor[4] and $R^{(2)} = [10000,0; 0,10000]$. As explained earlier, $H^{(i)} = I$ for $i = \{1,2\}$, that is, for both sensors. For the RFS-Combined-Map, $\theta_{combined} = 20$ is chosen.

During the matches, which were 10 minutes each, every robot played normally, applying the same rules of penalization as in a real RoboCup competition. The results of these matches can be seen in Table 1 (best, average, and worst OSPA errors for each case). The results show that the OSPA errors of the RFS-Local-Map and the RFS-Combined-Map methods are lower than the error of the MH-EKF method. Error reductions of 17.29 percent and 35.73 percent are obtained by RFS-Local-Map and the RFS-Combined-Map, respectively.

In the second set of experiments, 10 matches of 1 versus 1 were performed. In this case, only the RFS-Local-Map is compared against the MH-EKF method (there is no shared information to use). These experiments were performed to show that the proposed method overcomes the difficulties of data association and has a better performance even though no information is shared by other robots. As can be seen in Table 1, in this case, RFS-Local-Map obtains an error reduction of 32.62 percent when compared to the baseline method.

Our proposed methodology can integrate information shared by teammate robots, their positions, and their estimations of other robots' positions (that is, their local maps). We validated our methodology in several soccer matches and compared with a classical multihypothesis EKF tracking methodology. Although the specific application of the proposed tracking methodology is soccer robotics, its adaptation to other robot tracking applications/environments is straightforward. ■

THE AUTHORS

Pablo Cano is an MS student in the Department of Electrical Engineering at the AMTC Center, Universidad de Chile. His research interests include robotics, computer vision, and decision-making systems. Cano was a member of the UChile Robotics Team, which participates in the RoboCup Competition, between 2011 and 2016, leading the team in his last year. Contact him at pcano@die.uchile.cl.

Javier Ruiz-del-Solar is director of the Advanced Mining Technology Center at the AMTC Center, Universidad de Chile. His research interests include mobile robotics, autonomous systems, and human-robot interaction. Ruiz-del-Solar received a Doctor-Engineer from the Technical University of Berlin. He's recipient of the IEEE RAB Achievement Award 2003, RoboCup Engineering Challenge Award 2004, RoboCup @Home Innovation Award 2007, and RoboCup @Home Innovation Award 2008. Ruiz-del-Solar is a senior member of IEEE. Contact him at jruizd@ing.uchile.cl.

Acknowledgments

This work was partially funded by the Advanced Mining Technology Center of the Universidad de Chile and by FONDECYT Project 1161500.

References

1. T. Schmitt et al., "Probabilistic Vision-Based Opponent Tracking in Robot Soccer," *Robocup 2002: Robot Soccer World Cup VI*, LNAI 2752, Springer, 2003, pp. 426–434.
2. R. Marchant, P. Guerrero, and J. Ruiz-del-Solar, "Cooperative Global Tracking Using Multiple Sensors," *Robocup 2012: Robot Soccer World Cup XVI*, LNAI 7500, G.A. Kaminka et al., eds., Springer, 2013, pp. 310–321.
3. A. Fabisch, T. Laue, and T. Röfer, "Robot Recognition and Modeling in the RoboCup Standard Platform League," *Proc. 5th Workshop Humanoid Soccer Robots*, 2010, pp. 65–70.
4. J. Santos and P. Lima, "Multi-Robot Cooperative Object Localization: Decentralized Bayesian Approach," *RoboCup 2009: Robot Soccer World Cup XIII*, LNCS 5949, J. Baltes et al., eds., Springer, 2010, pp. 332–343.
5. I.R. Goodman, R.P.S. Mahler, and H.T. Nguyen, *Mathematics of Data Fusion*, Springer, 1997.
6. R.P.S. Mahler, "A Theoretical Foundation for the Stein-Winter 'Probability Hypothesis Density (PHD)' Multitarget Tracking Approach," *Proc. MSS Nat'l Symp. Sensor and Data Fusion*, vol. I, 2000, pp. 99–117.
7. R. Mahler, "A Brief Survey of Advances in Random-Set Fusion," *Proc. Int'l Conf. Control, Automation and Information Soc.*, 2015, pp. 62–67.
8. P. Dames and V. Kumar, "Autonomous Localization of an Unknown Number of Targets without Data Association Using Teams of Mobile Sensors," *IEEE Trans. Automation Science and Engineering*, vol. 12, no. 3, 2015, pp. 850–864.
9. P. Dames, P. Tokekar, and V. Kumar, "Detecting, Localizing, and Tracking an Unknown Number of Moving Targets Using a Team of Mobile Robots," *Proc. Int'l Symp. Robotics Research,* 2015, pp. 513–529.
10. D. Moratuwage, B.-N. Vo, and D. Wang, "Collaborative Multi-Vehicle SLAM with Moving Object Tracking," *Proc. IEEE Int'l Conf. Robotics and Automation*, 2013, pp. 5702–5708.
11. R.P.S. Mahler, *Statistical Multisource-Multitarget Information Fusion,* Artech House, 2007.
12. B.-N. Vo and W.-K. Ma, "The Gaussian Mixture Probability Hypothesis Density Filter," *IEEE Trans. Signal Process.*, vol. 54, no. 11, 2006, pp. 4091–4104.
13. R. Mahler, "The Multisensor PHD Filter, I: General Solution via Multitarget Calculus," *Proc. SPIE,* 2009, pp. 73360E–73360E–12.
14. J. Mullane et al., *Random Finite Sets for Robot Mapping and SLAM,* Springer, 2011.
15. G. Jochmann et al., "Efficient Multi-Hypotheses Unscented Kalman Filtering for Robust Localization," *RoboCup 2011: Robot Soccer World Cup XV*, LNCS 7416, T. Röfer et al., eds., Springer, 2011, pp. 222–233.
16. T. Laue and T. Röfer, "Simrobot-Development and Applications," *Proc. Int'l Conf. Simulation, Modeling and Programming for Autonomous Robots*, 2008, pp. 143–150.
17. D. Schuhmacher, B.-T. Vo, and B. Vo, "A Consistent Metric for Performance Evaluation of Multi-Object Filters," *IEEE Trans. Signal Processing*, vol. 56, no. 8, 2008, pp. 3447–3457.