Multi-Target & Multi-Detector People Tracker for Mobile Robots

Andreu Corominas-Murtra[1,2]
[1]Beta Robots
Barcelona, Catalunya, Spain
www.beta-robots.com
[2]Institut de Robòtica i Informàtica Industrial (CSIC-UPC)
Barcelona, Catalunya, Spain
www.iri.upc.edu

Jordi Pagès[3]
[3]PAL Robotics
Barcelona, Catalunya, Spain
www.pal-robotics.com

Sammy Pfeiffer[3]
[3]PAL Robotics
Barcelona, Catalunya, Spain
www.pal-robotics.com

***Abstract*—People tracking is a key perception skill for mobile robots designed to share environments with human beings. It allows the robot to keep track of people around them, which is fundamental for two main reasons: safety and social interaction. This paper presents the work done on people tracking with the REEM robot after two years of paticipation at the RoboCup@home challenge. The main contribution of the paper is the tracker part, which is designed to be multi-target and to fuse heterogeneous detections from a variety of sensors, each one yielding different rates, field of views and quality performance. The paper carefully describes the tracker approach, based on multi-target particle filtering, as well as data association step, based on a probabilistic multi-hypothesis tree. Quantitative evaluations of real datasets using CLEAR MOT metrics are provided, comparing different sensor/detector set-ups and different data association approaches.**

I. Introduction

Mobile robots are already sharing environments with humans in real indoor and outdoor scenarios like museums [1], offices [2] or pedestrian areas [3]. In all situations, such robots need to keep track of people present around it, specially for two main purposes: safety and social interaction. Safety in terms of avoiding collisions with humans, and social interaction with the aim of stablishing relations such as conversation, guiding or following.

In this paper we focus on the perception challenge of keeping track of positions of people around, from a mobile robot. This tracking skill gets inputs from a set of detectors, ranging from full body detections to body parts ones, such as legs or faces, and may involve multi-sensor fusion. The tracker works on a robot centered coordinate frame, and its goal is to estimate the most likely position for each target, given the detections, while correctly associating these detections with current tracks for a proper estimation.

People tracking from a moving platform has attracted the attention of the research community due to its large variety of applications in robotics and in automated or assisting driving systems for cars. The main differences among the existing works are three-fold: a) The type of sensors and detectors used, which determine the type of information regarding the people location, its accuracy and its reliability. b) The tracking algorithm which includes the state space definition, motion models and estimates the position of the tracked people. c) The data association techniques to match detections with tracks.

Fig. 1. REEM robot at FollowMe test. Robocup@home'13, Eindhoven.

Regarding the **type of sensors** involved, there are works which use sonar rings [4], laser scanners [5], [6], monocular cameras [7], [8], stereo cameras [9], [10], [11], thermal cameras [12], RGB-D cameras [13], [14], [8], [15], [16] and others addressing data fusion of a combination of some of these sensors [4], [17], [12]. Different **tracking algorithms** can be found in the literature. In [9] *alpha-beta* filters are used. Variants of Kalman filtering are pretty common. The regular *Kalman Filter* is used in [13], while in [15] authors rely on the *Extended Kalman Filter*. The *Unscented Kalman Filter* is adopted in [17], [14]. Finally, *Particle Filters* are also widely used by the community [5], [12], [8]. Among common **data association** techniques some authors adopt *Nearest Neighbor* [17], [11], [14], [16], which can be based on the Euclidean distance or some defined likelihood. *Multi-hypothesis tracking*, based on evaluating the tree of all possible associations, was initially proposed by [18], and used in the mobile robotics domain by [6], [13]. Even if it has exponential cost which may prevent real-time use when the number of tracks increase. An algorithm able to overcome this problem is the *Joint Probabilistic Data Association* [19], but it has the drawback that the number of objects to track must be known. In [5] this limitation is addressed and a way to estimate the number of objects is proposed along with an adaptation of the JPDA. In recent years hypothesis selection frameworks based on *Minimum Description Length* have been also studied [10], [20], [15].

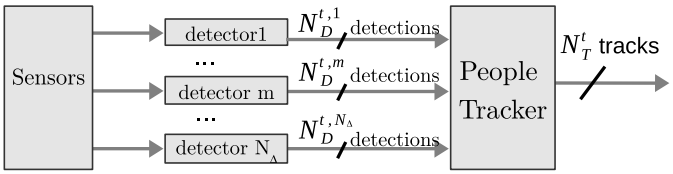

Fig. 2. Multi-target & multi-detector people tracking approach.

This paper presents a fully integrated work on people tracking developed during two years of participation at the RoboCup@home challenge [21] with REEM robot [22] shown in Fig. 1. The main contribution is the tracker part, which is designed to be multi-target and to fuse heterogeneous detections from a variety of detectors, each one yielding different rates, field of views and quality performance. The tracker architecture is completely ready to fuse more detectors in a standarized way and dedicates a particle filter for each target. In present implementation, detectors and tracker run in a standard computer, the later achieving rates up to 30 Hz. Tracker code is available in a public repository[1], fully ROS integrated [23] and licensed under BSD terms.

Section II of this paper presents the problem and Section III formalises the tracker in a general way, which is the main contribution of this work. Section IV decribes the experimental environment and provides quantitative results about the tracker performance following the CLEAR MOT metrics [24] in real world datasets. Finally, section V discusses the main conclusions of this work.

II. Problem Description

From the point of view of a mobile robot, people tracking skill can be summarized as the ability to keep track of humans around the robot, which implies an accurate estimation of person positions, as well as a robust association of targets with the same ID through time.

People tracking is usually thought as a multi-target tracking, sensor fusion algorithm, which receives inputs of a set of heterogeneous and asynchronous detectors and provides people tracks at a constant rate. Given a set of detectors $\Delta = \{\Delta^1, ..., \Delta^m, ...\Delta^{N_\Delta}\}$ running at different rates, at a given tracker iteration t the tracker receives a set of detections, $D^t = \{D_i^{t,m}\}$, where index t is the iteration counter, index m the detector counter and index i runs over detections. At each iteration t, the tracker outputs a set of tracks $T^t = \{T_1^t, ...T_j^t, ..., T_{N_T^t}^t\}$, where index j runs over the target list. Therefore, at each iteration there are $N_D^{t,m}$ detections for each detector m, and N_T^t targets being tracked. For all detectors ($\forall m \in [1, N_\Delta]$), at a given iteration t, there is no assumption on the relation between $N_D^{t,m}$ and N_T^t, so they both could range from 0 to any reasonable positive integer (see Fig. 2).

For effective sensor fusion, all position and velocity components through the work are referenced to a coordinate frame centered on the robot, extending along the horizontal plane, with its X axis aligned with robot forward axis and its Y axis aligned with robot left axis. This step implies a full calibration of all sensor mounting points involved.

[1]https://github.com/beta-robots/pipol_tracker

III. Tracker

In the tracker side, estimation and fusion is based on particle filtering, while data association is implemented by a probabilistic tree, which is built for each detector.

State components of particles lie in $\mathbf{s} = (\mathbf{x}, \mathbf{v}) = (x, y, u, v)$, being $\mathbf{x} = (x, y)$ the target position and $\mathbf{v} = (u, v)$ its linear velocities along X and Y axis of robot frame respectively (horizontal plane). At iteration t, the n^{th} particle of the j^{th} target is expressed as:

$$\mathbf{a}_j^{t,n} = \{\mathbf{s}_j^{t,n}, w_j^{t,n}\} = \{(x_j^{t,n}, y_j^{t,n}, u_j^{t,n}, v_j^{t,n}), w_j^{t,n}\}, \quad (1)$$

where $w_j^{t,n} \in [0, 1]$ is the particle weight. Each single target is supported by a constant amount of particles N_S. The current estimate of target j is expressed as $\mathbf{s}_j^t = (\mathbf{x}_j^t, \mathbf{v}_j^t)$. Depending on the context within the iteration, this estimate, as well as the supporting particle set, can be viewed as a *prior* or a *posterior*. Algorithm 1 outlines a single tracker iteration. The following subsections explain in detail each step of the algorithm.

Algorithm 1 Tracker Iteration

INPUTS: $T^{t-1}, D^t, \mathbf{o}^t$
OUTPUT: T^t
$particlePrediction(), \ \forall j, n$ //people walking
$particlePrediction(\mathbf{o}^t), \ \forall j, n$ //robot odometry
$occlusionPrediction()$
$dataAssociation(D^{t,m}, T^t), \ \forall m$
$correction(D^t) \ \forall j, n$
$resampling() \ \forall j, n$
return T^t

A. Prediction

Prediction step is divided according to two main motion events occurred within an iteration time step, τ: people walking and robot odometry.

People Walking: is solved through propagating each particle state according a linear and constant velocity model, with addition of synthetic noise to account for unmodelled aspects $\sigma_\mathbf{x}, \sigma_\mathbf{v}$. The model is summarized as:

$$\mathbf{s}_j^{t,n} = \begin{bmatrix} 1 & 0 & \tau & 0 \\ 0 & 1 & 0 & \tau \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{s}_j^{t-1,n} + \begin{bmatrix} \sigma_\mathbf{x} \\ \sigma_\mathbf{x} \\ \sigma_\mathbf{v} \\ \sigma_\mathbf{v} \end{bmatrix} \quad (2)$$

Robot Odometry: Robot travel is assumed as an input provided by some odometry source, $\mathbf{o}^t = (r^t, \theta^t)$. This odometry increment causes apparent motion of all tracked targets, since the tracker works on robot centered coordinates. Thus, all particles have to be moved according to the odometry model described by the following equations:

$$\mathbf{H}_n^{t,t-1}(\mathbf{o}^t) = \mathbf{H}_n^t = \begin{bmatrix} \cos\theta_n^t & -\sin\theta_n^t & r_n^t \cos\frac{\theta_n^t}{2} \\ \sin\theta_n^t & \cos\theta_n^t & r_n^t \sin\frac{\theta_n^t}{2} \\ 0 & 0 & 1 \end{bmatrix}, \quad (3)$$

which is the 2D homogeneous transformation that moves points from t to $t-1$. To manage model inaccuracies, odometry errors and unknown timing errors, for each particle we add

synthetic Gaussian noise to both odometry increments when computing $\mathbf{H}_n^t$. Therefore $\mathbf{H}_n^{t,t-1}$ has dependency on n index (particle index), and

$$r_n^t = r^t + \mathcal{N}(0, \sigma_r);\ \theta_n^t = \theta^t + \mathcal{N}(0, \sigma_\theta), \tag{4}$$

where $\mathcal{N}()$ is the added Gaussian noise. Since target points and velocities have to be moved from $t-1$ to t, we apply the following motion, $\forall j = 1..N_T^t, \forall n = 1..N_S$:

$$\begin{bmatrix} \mathbf{x} \\ 1 \\ \mathbf{v} \\ 0 \end{bmatrix}_{j,n}^{t} = \begin{bmatrix} (\mathbf{H}_n^t)^{-1} & \mathbf{0}_3 \\ \mathbf{0}_3 & (\mathbf{H}_n^t)^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ 1 \\ \mathbf{v} \\ 0 \end{bmatrix}_{j,n}^{t-1}. \tag{5}$$

B. Occlusions

Just after particle prediction, but before data association, occlusions between targets are computed, according to target predicted positions.

At each iteration t, occlusion model considers a ray between the robot center and each target T_j^t. If any target $T_k^t, \forall k \neq j$, is closer to that ray than ρ_p (radius of a person), then the target T_j^t is considered occluded.

This step allows to label each target as occluded or not. Occluded targets will be skipped in the data association step for those detectors providing bearing-only measurements over the XY plane, such as a body detector in a monocular image. Detectors providing range data on XY, or directly full 3D detections, do not take into account occlusions in their data association.

C. Data Association

Data association is the most critical step in such multi-target tracking processes. This step assumes a set of detections $D^t = \{D_i^{t,m}\}$ coming from different detectors, and a set of targets $T^t = \{T_j^t\}$ currently tracked by the process. Please remind that index m runs over detectors, index i runs over detections and index j over targets. Considering these inputs, data association should establish, for each detector, pairs between detections and targets. But this pairing decision has to take into account the fact that a detection can result unassociated due to the presence of a new target or a due to a false positive. Moreover a target can also result unassociated if none detection was due to its presence, which is a false negative case. Data association is solved independently for each detector, so in the following of the subsection index m will be omitted. Index t will be also skipped since this step is solved for each iteration without any dependency on previous iterations. The goal is to find the most likely *association event* which globally maximizes the event probability.

To solve this problem and be able to compute such probabilities, a **tree** is built. The tree grows downwards for each detection, while grows sideways for each target. The top node of the tree is called *root node*, while all nodes that finish the tree (they do not have lower nodes below) are called *leaf nodes*. A set of nodes, starting from a given leaf node up to the root node, following upwards tree links, is called a *branch*. A branch represents a full association event. Within the tree, detection index i runs from 1 to N_D, but target index j ranges from 0 to N_T. Target $j = 0$ represents the *void* target,

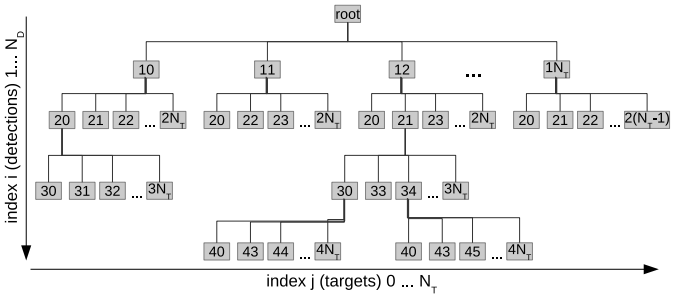

Fig. 3. Data association tree. The tree grows downwards driven by N_D detections, and sideways by $N_T + 1$ targets. Target $j = 0$ is the void target. In this figure, the tree is unfolded arbitrarily downwards due to space constraints of the paper, but horizontal unfolding shows how upper-level target indexes are skipped at each new down level, except for target $j = 0$.

allowing false positive detections or new target detections to be associated with it. Fig. 3 shows this tree. Please note that the tree grows downwards skipping previous targets already associated in the same branch, excepting target $j = 0$, which is always allowed to be associated.

Each node of the tree holds a probability p_{ij}, which describes the marginal probability that detection i is due to target j. Root node holds a $p_{00} = 1$. $\forall i = 1..N_D, j = 1..N_T$, probabilities p_{ij} are computed as follows:

$$p_{ij} = \lambda_{ij} \cdot (1 - p_{i0}) \cdot \prod_{\substack{k=1..N_T \\ k \neq j}} (1 - \lambda_{ik}) \cdot \prod_{\substack{k=1..N_D \\ k \neq i}} (1 - \lambda_{kj}), \tag{6}$$

while $\forall i = 1..N_D, j = 0$, expression for p_{i0} is:

$$p_{i0} = \prod_{\substack{k=1..N_D \\ k \neq 0}} (1 - \lambda_{ik}), \forall i = 1..N_D, \tag{7}$$

In both equations above, λ_{ij} terms are likelihood functions between detection i and target j, always designed to result in the range $[0, 1]$ (see subsection III-D). Therefore, the first term of equation 6 is the likelihood of the pair i, j, while the second term is the probability that the detection wasn't unassociated. The third term in equation 6 is the likelihood that detection i does not match other targets than j, while the last term is the likelihood that target j does not match other detections than i. Equation 7 only considers the likelihood that detection i does not match to actual targets, which is directly the probability of detection i being unassociated. Nodes that have p_{ij} below a certain threshold p_Z stop the tree growing. Once p_{ij} are computed, they need to be normalized. The sum of all p_{ij} for all nodes sharing the same upper node (same parent node) have to be 1.

After growing the tree and computing its probabilities, the most likely association event is choosen. At this stage, each leaf node starts a branch which finishes at the root node. Each of these branches defines a complete association event. Assuming that node probabilities are independent for a given branch, the selection of the most likely event requires computation of branch probabilities, which are directly the event probabilities. From each of the leaf nodes, the event probability P_k is computed by going up through the branch

and multiplying each p_{ij} contribution:

$$P_k = \prod_{\substack{\forall nodes \\ \in branch\ k}} p_{ij} \tag{8}$$

The branch with higher probability, named $\mathfrak{B}$, holds the most likely association event. Detection-target pairs can be directly extracted from nodes of this branch as the output of the data association step.

D. Correction

Correction step updates particle weights, $w_j^{t,n}$, $\forall j = 1..N_T^t, n = 1..N_S$. This correction is computed by evaluating a likelihood function for each particle (index n), for each target (index j), and for each available and associated detection (index i) at the current iteration. Such likelihood functions fulfill:

$$\lambda_{ijn}^m = \lambda(D_i^m, \mathbf{s}_j^n) : \mathbf{s} \rightarrow \mathbb{R} \in [0, 1] \tag{9}$$

These λ^m functions do not necessarily implement Gaussian-like likelihoods, and can be customized for each detector to fit better its statistical properties. For instance, a leg detector likelihood does not follow a Gaussian model since legs are usually out of the person center. Using particle filters lead our tracker approach flexible enough to deal with these kind of details, in contrast of KF-based implementations.

At the end of the correction step, fusion is finally implemented under the assumption of statistical independence between detectors:

$$w_j^{t,n} = \prod_{\substack{m=1..N_\Delta^t \\ (i,j)\in\mathfrak{B}}} \lambda_{ijn}^m, \quad \forall j = 1..N_T^t, \forall n = 1..N_S. \tag{10}$$

E. Resampling

A call to a resampling function is required in particle filtering to avoid particle depletion, which could cause that the filter no longer *explores* the state space appropiately [25]. Therefore, at the end of each iteration, a new particle set is drawn, based on the density represented by the current one. Moreover, synthetic noise is added to each particle state components, to assure a proper exploration of the state space.

F. Target Management

Targets are created for each unassociated detection after data association step. However, these targets start with a status of *candidate*. This early status is upgraded when the target is further associated in next iterations, passing through *legged* and *visually confirmed* up to *friend*. Each status upgrade has requirements in terms of a minimum number of detections associated to the target. At tracker output, only viusally confirmed and friend targets are published.

Target removal is also based on heuristics about a maximum number of iterations allowed without detections being associated. Depending on the status of the target, this threshold can be greater, so removing a friend target is harder than removing a candidate one.

IV. Experimental Results

This section overviews the experimental environment used to extract the performance results of our tracker approach. For qualitative results, please refer to figure 4 or see the publicly available video at [2].

A. Sensors and Detectors

The presented tracking approach is general in terms of which detectors can be fused. However, in order to experiment and evaluate our tracking approach, we use a set of 4 detectors available on the REEM robot [22].

The first one is a **leg detector** fed by Hokuyo laser scanner measurements, inspired by the solution proposed in [26]. This detector spans over an area of 180^o, we consider detections up to 5m, and it runs at $12Hz$. The second detector is a **body detector** receiving monocular images as input, which provides bearing-only measurements of the central point of the detected person at $6Hz$. The authors trained a Support Vector Machine with a Histogram of Oriented Gradients (HOG), which mainly captures how the silhouette of a person looks like. The 2D person detector used in this work is inspired by [27] and it makes use of the RGB camera of the Asus Xtion. RGB data from the Asus Xtion camera is also used by the third detector, which is a comercial **face detector** providing full 3D measurements of the central point of detected faces, up to 1.5m, at a rate of $6Hz$. Finally, a fourth detector is the **skeleton detector** included in the NiTE middleware (PrimeSense), which uses depth data from the Asus camera to provide quite reliable 3D detections of persons at $30Hz$. The algorithm is able to identify blobs in the depth image that correspond to persons. The centroid of every person blob is used to estimate the 3D position from the camera intrinsics and the depth value. Figure 5 shows a snapshot for each of these four detectors. As propioceptive data, wheel odometry provided by wheel encoders is also used as described in subsection III-A.

[2] www.youtube.com/watch?v=CrNyZsCSReM

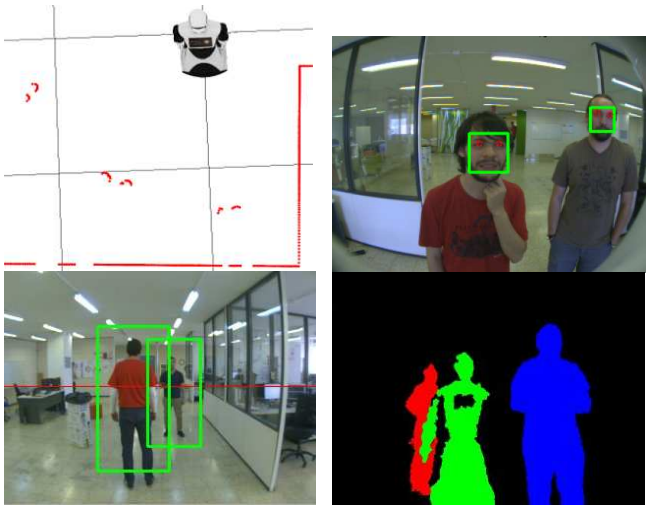

Fig. 5. Examples of detector outputs. Top-left: Clusters of points (in red) from a laser range scan showing typical human leg profiles. Top-right: Face detector. Bottom-left: Monocular body detector. Bottom-right: Skeleton detector.

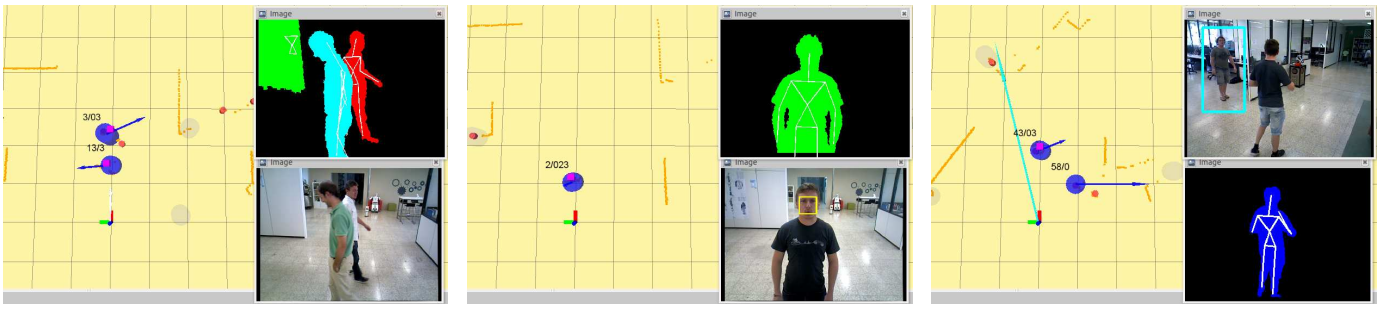

Fig. 4. Three real situations managed by our tracker in different executions. Big blue cylinders are *visually confirmed* or *friend* targets, while big grey cylinders are candidates. Small red cylinders are leg detections. Small magenta cubes are body3D detections, which are also shown in the image of coloured silhouettes. Cyan arrows are bearing 2D body detections, which are also drawn as bounding boxes in the RGB image, where yellow rectangles mark face detections too. The first number near to each target is its ID, while the numbers after the slash indicate which detectors are actually fused in the current iteration (0:leg, 1:body2D, 2:face, 3:body3D). First picture shows a crossing, with some false positive detections from leg and skeleton detectors. Second situation illustrates a face detection fused with legs and body3D. In the last case, one of the targets (with id=58) is out of the field of view of the camera, but it is still tracked thanks to the legs detector. Moreover a new target has recently appeared and it will be soon upgraded to visually confirmed status thanks to body2D bearing detections, even if it is not yet seen by the skeleton detector.

B. Software implementation

A solution of the presented tracker approach has been implemented as a C++ library, wrapped with a ROS-catkin node. The code is available in a GitHub public repository (see footnote, page 2). Library dependencies are OpenCV and Eigen, and it is licensed under BSD terms. Detectors are also wrapped as ROS nodes, so the tracker node subscribes separately to each detector main topic. The tracker outputs a set of tracked targets in a customized message and a standard marker array message for visualization purposes. To get the results presented below the tracker was set to run at $20Hz$, and the number of particles per target was $N_S = 500$. With this configuration, the tracker always used less than 60% of a single CPU of an Intel(R) Core(TM) i5-2410M CPU @ 2.30GHz processor.

C. Datasets and Ground Truth

Three datasets, named A, B and C, are used to evaluate different aspects of the tracking performance. All of them are recorded as rosbags from the REEM robot, they provide both raw and detection-level data, and each one lasts for around $60s$. They mainly hold typical situations such as crossings, stop and go, out of field of view, as well as different robot motions. Authors are available to share (upon request) these rosbags with interested research teams. These three datasets have been manually annotated to generate a ground truth, thanks to a developed software tool which uses ROS *interactive markers*. This tool is also available in the same public package mentioned above. Ground truth data has been annotated at $1Hz$, 180^o field of view and up to $5m$ range.

D. Performance Evaluation

In order to evaluate the performance of the proposed tracking approach, the CLEAR MOT metrics [24] have been used, which is specially designed for multi-target tracking problems and decouples the precision problem (MOTP index, meters) from the association problem (MOTA index, ratio). The experiments want to evaluate how the approach behaves in terms of *sensor fusion* and in terms of *data association*, so we worked on two main experimental scenarios. In both scenarios, each one of the below described configurations has been executed 10 times and the MOTP/MOTA entries presented in the tables below are the mean values of these 10 executions.

Sensor Fusion: performance is evaluated by comparing executions of the tracker while using all 4 detectors described in subsection IV-A, against executions using only a subset of such detectors. This experiment wants to show how sensor fusion improves the final performance of the tracker. Table I shows the CLEAR MOT results for each data set in three situations: ALL) fusing all 4 detectors, LBF) fusing only legs, body and face detectors and LS) fusing only legs and skeleton detectors. Regarding the MOTA index, these experimental results indicate how tracker performance gets worse in the LBF case, but is similar for ALL and LS configurations. One possible reason of that is the high rate of skeleton detector ($30Hz$) with respect to the others ($12Hz$, $6Hz$), so this detector plays a more important role in the final performance of the tracker. In terms of MOTP, all configurations have an overall precision of about $15cm$. In that case, leg detector, which is fed by laser range measurements, seems to play the major role.

TABLE I. CLEAR MOT METRICS IN SENSOR FUSION SCENARIO

	Dataset A			Dataset B			Dataset C		
	ALL	LBF	LS	ALL	LBF	LS	ALL	LBF	LS
MOTP	0.14	0.11	0.15	0.15	0.18	0.14	0.16	0.14	0.16
MOTA	0.69	0.63	0.71	0.31	0.15	0.32	0.66	0.32	0.61

Data association: performance is evaluated comparing the proposed multi-hypothesis tree (MHT), described in subsection III-C, against the classical nearest neighbor (NN) approach with Euclidean distances. Table II shows the CLEAR MOT metrics for each data set when using each data association method. This table shows how MHT improves MOTA index in all cases A, B and C, without observing an increase of CPU resources. Specifically, MHT solves better some crossing situations as well as some associations of false positives, specially coming from leg detector.

TABLE II. CLEAR MOT METRICS IN DATA ASSOCIATION SCENARIO

	Dataset A		Dataset B		Dataset C	
	MHT	NN	MHT	NN	MHT	NN
MOTP	0.14	0.15	0.15	0.15	0.16	0.15
MOTA	0.69	0.34	0.31	0.21	0.66	0.56

V. CONCLUSIONS

This paper has reported the work done on people tracking with the REEM robot after two years of paticipation at the RoboCup@home challenge. The main contribution of the paper is the tracker part, which is designed to be multi-target and to fuse heterogeneous data provided by a variety of detectors, each one yielding different rates, field of views and quality performance. The multi-detector approach is a promising solution, since it provides robustness against single detector false positives, as well as increase the field of view of a single sensor. The multi-hypothesis tree technique also described in the paper clearly improves MOTA metrics in the presented experimental results. Moreover, the presented tracker method, as well as the published C++ code, is specially thought to decouple detectors from tracker, so integration of new detectors to the tracker framework is done in a standarized way, mainly by designing a new λ^m likelihood function, which is used in two key steps of the tracker: data association and particle correction.

Future works will face four main directions: 1) improving people motion models, 2) include social constraints between tracked people, 3) extracting and tracking target appearance parameters, and 4) investigate alternative state estimation frameworks beyond Kalman or Particle filtering.

ACKNOWLEDGMENT

The authors would like to thank to all REEM@IRI and REEM@LaSalle team members for their work on the RoboCup@home challenges held in 2013 and 2014.

REFERENCES

[1] W. Burgard, A. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun, "Experiences with an interactive museum tour-guide robot ," *Artificial Intelligence*, vol. 114, no. 1-2, 1999.

[2] E. Marder-Eppstein, E. Berger, T. Foote, B. Gerkey, and K. Konolige, "The Office Marathon: Robust Navigation in an Indoor Office Environment," in *Proceedings of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, Anchorage, USA. May, 2010.

[3] E. Trulls, A. Corominas Murtra, J. Pérez-Ibarz, G. Ferrer, D. Vasquez, J. M. Mirats-Tur, and A. Sanfeliu, "Autonomous navigation for mobile service robots in urban pedestrian environments," *Journal of Field Robotics*, vol. 28, no. 3, 2011.

[4] T. Wilhelm, H. J. Böhme, and H. M. Gross, "Sensor fusion for vision and sonar based people tracking on a mobile service robot," in *Proc. Int. Workshop on Dynamic Perception*, 2002.

[5] D. Schulz, W. Burgard, D. Fox, and A. Cremers, "Tracking multiple moving targets with a mobile robot using particle filters and statistical data association," in *Proc. IEEE Int. Conf. on Computer Vision & Pattern Recognition*, vol. 1, Kauai, USA, 2001.

[6] M. Luber, G. Diego, T. Kai, and K. Arras, "Spatially grounded multi-hypothesis tracking of people," in *Proc. IEEE Int. Conf. on Robotics and Automation*, Kobe, Japan, 2009.

[7] A. Mykhaylo, S. Roth, and B. Schiele, "Monocular 3d pose estimation and tracking by detection," in *Proc. IEEE Int. Conf. on Computer Vision & Pattern Recognition*, 2010.

[8] W. Choi, C. Pantofaru, and S. Savarese, "A general framework for tracking multiple people from a moving camera," *Pattern Analysis and Machine Intelligence*, 2013.

[9] D. M. Gavrila and S. Munder, "Multi-cue pedestrian detection and tracking from a moving vehicle," *Int. Journal of Computer Vision*, vol. 73, no. 1, 2007.

[10] B. Leibe, K. Schindler, N. Cornelis, and L. Van Gool, "Coupled object detection and tracking from static cameras and moving vehicles," *IEEE Tr. Pattern Analysis and Machine Intelligence*, vol. 30, no. 10, 2008.

[11] M. Bajracharya, B. Moghaddam, A. Howard, S. Brennan, and L. H. Matthies, "Results from a real-time stereo-based pedestrian detection system on a moving vehicle," in *Proc. IEEE Int. Conf. on Robotics and Automation*, Kobe, Japan, 2009.

[12] G. Cielniak, T. Duckett, and A. J. Lilienthal, "Improved data association and occlusion handling for vision-based people tracking by mobile robots." in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2007.

[13] M. Luber, L. Spinello, and K. O. Arras, "People tracking in rgb-d data with on-line boosted target models," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2011.

[14] F. Basso, M. Munaro, S. Michieletto, E. Pagello, and E. Menegatti, "Fast and robust multi-people tracking from rgb-d data for a mobile robot," *Advances in Intelligent Systems and Computing*, vol. 193, no. 1, 2012.

[15] O. H. Jafari, D. Mitzel, and B. Leibe, "Real-time rgb-d based people detection and tracking for mobile robots and head-worn cameras," in *Proc. IEEE Int. Conf. on Robotics and Autom.*, Hong Kong, 2014.

[16] M. Munaro and E. Menegatti, "Fast RGB-D People Tracking for Service Robots," *Journal on Autonomous Robots, Springer*, 2014.

[17] N. Bellotto and H. Hu, "Multisensor-based human detection and tracking for mobile service robots," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 39, no. 1, 2009.

[18] D. Reid, "An algorithm for tracking multiple targets," *IEEE Transactions on Automatic Control*, vol. 24, no. 6, pp. 843–854, 1979.

[19] I. Cox, "A review of statistical data association techniques for motion correspondence," *Int. Journal of Computer Vision*, vol. 1, no. 10, 1993.

[20] B. Leibe, A. Leonardis, and B. Schiele, "Robust object detection with interleaved categorization and segmentation," *Int. Journal of Computer Vision*, vol. 1-3, no. 77, 2008.

[21] T. Wisspeintner, T. van der Zant, L. Iocchi, and S. Schiffer, "RoboCup@Home: Scientific competition and benchmarking for domestic service robots," *Interaction Studies*, vol. 10, no. 3, 2009.

[22] L. Marchionni, J. Pagès, J. Adell, J. R. Capriles, and H. Tomé, "Reem service robot: how may i help you?" in *Int. Work-Conf. on the Interplay between Natural and Artifical Computation*, Mallorca, Spain, 2013.

[23] Open Source Robotics Foundation, "ROS website," `www.ros.org`.

[24] K. Bernardin and R. Stiefelhagen, "Evaluating Multiple Object Tracking Performance: The CLEAR MOT Metrics," *EURASIP Journal on Image and Video Processing*, 2008.

[25] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, " A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking," *Tr. on Signal Processing*, vol. 50, no. 2, 2002.

[26] K. Arras, O. Mozos, and W. Burgard, "Using boosted features for the detection of people in 2d range data," in *Proc. IEEE int. Conf. on Robotics and Automation*, Roma, Italy, 2007.

[27] S. Paisitkriangkrai, C. Shen, and J. Zhang, "Real-time pedestrian detection using a boosted multi-layer classifier," in *8th IEEE Int. Workshop on Visual Surveillance, in conjunction with European Conf. on Computer Vision*, Marseille, France, 2008.