Knowledge-Driven World Modeling

J. Elfring, S. van den Dries, M.J.G. van de Molengraft and H. Bruyninckx

Abstract— **In order to allow safe operation and good performance, robots need an accurate model of their environment. Such a world model is constructed from sensor detections and typically contains information about positions and velocities of surrounding objects. This paper proposes how prior knowledge about those objects can be used to improve the performance of a world model, which is implemented by a Multiple Hypothesis Filter (MHF). More specifically, knowledge about object dynamics, expected locations, relations between object classes and detector characteristics is incorporated in the probabilistic models of the MHF. The results of simulations confirmed the potential of incorporating such object knowledge in a world model.**

I. INTRODUCTION

In order to allow safe operation and good performance, robots need an accurate model of their environment. This view on the world will be called world model and in this paper it contains both the positions and velocities of a set of semantically labeled objects. Typically, the world model's goal is to enable safe navigation and successful manipulation of objects. Furthermore, it can take load off perception by providing expected object locations.

An important prerequisite for such a world model is a real-time applicable multiple sensor multiple target data association approach that can operate in unstructured and dynamically changing environments. The data association determines whether an observed feature corresponds to a particular object in the world model. Target tracking allows filtering out measurement noise, estimating velocities and propagating object positions, even if they are out of sight, *e.g.*, temporarily occluded.

Various algorithms aim at solving the data association problem. The joint probabilistic data association filter [1], the Multiple Hypothesis Filter (MHF) [2] and the probability hypothesis density filter [3] are most widely used as a basis. For a more complete overview see [4]. All the state of the art extensions use target tracking. Based on previous work [5], the MHF is selected as being the preferred solution for the data association problem.

The basic idea underlying a MHF is to keep track of all possible solutions to the data association problem. Each solution is represented by one hypothesis and the total set of solutions forms a hypothesis tree. At each time step, the state of the world according to the most probable hypothesis is used as a world model. However, the size of the tree grows more than exponentially with the number of measurements rendering the algorithm computationally intractable. In order to allow real-time operation, each time step, a subset of possible solutions is pruned based on the probability that the hypothesis is the correct one. As a result, there is no longer a guarantee that the most probable solution is available at any time step, since an ancestor of the most probable hypothesis may have been pruned at an earlier time step. This means that the quality of the world model is highly dependent on the probabilistic models underlying the calculation of hypothesis probabilities. Object specific knowledge, such as 'tables are static objects' or 'people enter through a door', can be of great help by calculating the probability of a hypothesis. Furthermore, it seems a missed opportunity to not use prior knowledge about the world.

Various knowledge based approaches exist that solve the data association problem based on object knowledge. There, the problem of data association is also referred to as object identity resolution [6]. Knowledge bases are used to allow reasoning, *e.g.*, visual models of objects available in online databases such as [7] – [12], or knowledge bases representing common sense knowledge or relations between objects, *e.g.*, [13] – [15]. According to [6], object identity resolution 'has not received substantial attention' and due to their 'fairly low' update rates, tracking as used in the MHF is not a helpful option. However, experiences, *e.g.*, in the RoboCup middle size league where robots move up to 5 [m/s] [5], have clearly shown the need for object tracking, especially in dynamical environments. Furthermore, tracking enables predicting object positions, even if they are temporarily occluded and it provides regions of interest to perception, which makes object detection cheaper, thereby allowing for higher update rates. A more related approach is the perceptual anchoring in [16], however, their experiments focus on reasoning with static objects.

The aim of this paper is to incorporate object knowledge and use it to improve the performance of the MHF by designing better probabilistic models. A side contribution is the introduction of a strategy that removes objects from a hypothesis tree. The latter is needed, since a side-effect of the pruning is that once all hypotheses contain the same object, it will stay in the world model forever, even when this object has left the robot's environment. Summarized, the contributions of this paper are:

- Incorporation of object specific knowledge in the probabilistic models underlying the MHF. This way, object knowledge improves the performance under the constraint of a real-time applicability.
- Extension of the MHF with a mechanism that compen-

J. Elfring, S van den Dries and M.J.G. van de Molengraft are with the Department of Mechanical Engineering, Eindhoven University of Technology, The Netherlands. {J.Elfring, S.v.d.Dries}@tue.nl

H. Bruyninckx is with the Department of Mechanical Engineering, Katholieke Universiteit Leuven, Belgium

sates for leaving objects based on object models.

The paper is organized as follows. Section II introduces mathematical symbols and definitions and Section III explains the multiple target tracking approach that is used for solving the data association problem. Then, Section IV focuses on incorporating object knowledge in this framework and zooms in on the implementation. Section V presents simulation results, whereas Section VI gives conclusions and recommendations for future work.

II. MATHEMATICAL SYMBOLS AND DEFINITIONS

This section introduces the notation and definitions as used throughout this paper.

Each possible solution to the data association problem is represented by a hypothesis. A hypothesis is written as h_k^i, where the superscript i is the index of the hypothesis at discrete time step k. Each hypothesis exclusively contains a list of objects, where an object is represented by o. Each object has a semantic class label $C(o)$, such as 'Cup' or 'Person', and a unique identifier which allows distinction between objects of the same class. The class labels are hierarchical ordered in a taxonomy, which represents is-a relations between the classes. In addition, each object o has a Kalman filter, used for tracking the object. The filter has a state x_k, which at least contains a 3D position in case of static objects, and both a 3D position and a 3D velocity in case of non-static objects. The filter is updated based on measurements z_k, which represents the class label and position of an observed object. The class label of measurement z_k is denoted by $C(z_k)$.

III. TRACKING AND DATA ASSOCIATION

This section briefly explains the algorithm used for object tracking and data association. The algorithm takes measurements from a perception module as input and creates a world model out of it.

A. Expanding the Tree

Each time a measurement arrives, the tree is expanded. All possible solutions are considered, hence for each hypothesis the new measurement could be:

1) Clutter (a false detection)
2) An object that was already present in the parent hypothesis
3) A newly appeared object

If a measurement is associated with a new object, the object receives its own filter with the observed position and zero velocity as initial state. In [5], stationary Kalman filters were used but these filters can equally well be selected differently based on, *e.g.*, the dynamic properties of objects that are involved. Each hypothesis is unique, *i.e.*, the number of objects or the state of at least one of the objects will be different, and does not explicitly contain the association history.

B. Propagating Objects

Once the tree is expanded, all objects in all hypotheses are propagated. This means, that based on the system model and x_{k-1}, a prediction x_k of the real object state is made.

If the measurement is associated with an object, the object state propagation is followed by a state update based on this measurement.

C. Updating Probabilities

The posterior probabilities of the hypotheses are updated using Bayes law:

$$p(h_k^i|z_{1:k}) = \frac{p(z_k|h_k^i)p(h_k^i|h_{k-1}^{i'})p(h_{k-1}^{i'}|z_{1:k-1})}{p(z_k|z_{1:k-1})}, \quad (1)$$

where $h_{k-1}^{i'}$ represents the parent hypothesis of h_k^i and

$p(h_k^i|z_{1:k})$*:* is the posterior probability of the i'th hypothesis given all measurements up to and including timestep k.

$p(z_k|h_k^i)$*:* is the likelihood of measurement z_k given hypothesis h_k^i.

In case of associating z_k with clutter, *i.e.*, z_k is a false detection, $p(z_k|h_k^i)$ typically is a uniform distribution over the observation space.

In case of associating z_k with an existing object, the mismatch between the updated 3D object position according to the filter and the 3D object position according to the measurement is defined by the vector δ. The vector δ is transformed into a probability using a scaled multivariate normal distribution:

$$p(z_k|h_k^i) = e^{-\frac{1}{2}\delta^\top \Sigma^{-1}\delta}, \quad (2)$$

where Σ is the covariance matrix and typically represents the measurement noise.

In case of associating z_k with a new object $p(z_k|h_k^i) = 1$, since then the filter is initialized at the measured position and, as a result, δ is the zero vector.

$p(h_k^i|h_{k-1}^{i'})$*:* represents the conditional probability of the measurement being (i) a newly appeared object, (ii) an object that was already present in the hypothesis, or (iii) clutter, given its parent $h_{k-1}^{i'}$.

$p(h_{k-1}^{i'}|z_{1:k-1})$*:* represents the posterior probability of the parent hypothesis.

$p(z_k|z_{1:k-1})$*:* is the marginal probability that acts as a normalizing factor.

The pruning of hypotheses and the selection of the final hypothesis will completely be based on the probabilities calculated by (1), as will be shown in the next sections. For that reason, the likelihood $p(z_k|h_k^i)$ and the probabilistic models underlying $p(h_k^i|h_{k-1}^{i'})$ are crucial for the quality of world model compared to the ground truth.

D. Pruning of Hypotheses

In order to allow real-time implementation of the MHF, the size of the tree must be limited, which requires pruning of hypotheses. At each time step, two criteria are evaluated and all hypotheses that do not meet these criteria are removed

from the tree. The assumption underlying this pruning is that once the probability of a hypothesis (i) drops below a certain threshold or (ii) is no longer among the set of $n_{hyp,max}$ highest probabilities, it will never evolve to the hypothesis that best represents the state of the world in the future. More specifically, hypotheses are pruned if:

1) The probability $p(h_k^i|z_{1:k})$ of the hypothesis drops below a certain threshold:

$$\alpha \cdot \max_i p(h_k^i|z_k), \quad 0 \leq \alpha < 1, \tag{3}$$

where α describes the ratio between the maximum and minimum allowed hypothesis probability.

2) The total number of hypotheses exceeds a predefined number $n_{hyp,max}$ and h_k^i is not among the $n_{hyp,max}$ most probable hypotheses.

E. Removing Leaving Objects From Hypotheses

As a result of the hypotheses pruning, an object can be present in all $n_{hyp,max}$ most probable hypotheses in the tree. In this case the object will stay in the world model forever, even when it leaves the robot's environment and does no longer produce features. In literature, this problem is solved by removing objects from a hypothesis if the time since the last filter update exceeds a predefined threshold [5], [17], [18].

Some object removal strategy obviously is required, but if objects go out of sight, due to occlusion or movements of the robot, it usually is better to remember them. Furthermore, not detecting an object could also be the result of the perception module giving priority to detecting other objects, *e.g.*, detecting static objects like tables might only happen at very low rates. Here it is proposed to prune objects based on the uncertainty of the object state. More specifically, objects are removed from all hypotheses if the uncertainty, here represented by their covariance, exceeds a predefined threshold P_{max}.

F. Publishing the Result

Finally, at each time instance k the current state of the world according to the most probable a posteriori hypothesis h_k^{MAP} is given as an output of the algorithm.

IV. OBJECT KNOWLEDGE

Section III has clearly shown the importance of the probabilistic models, since these determine the most probable hypothesis. Furthermore, the pruning strategy is completely based on the probabilities $p(h_k^i|z_{1:k})$ that are calculated using these models. Although it seems beneficial to use prior object knowledge within the probabilistic models, this field of research is largely unexplored. For that reason, this section proposes how knowledge about objects can be used to improve tracking and data association. Section IV-A aims at explaining which object knowledge can be used and how. Then Section IV-B focuses on the implementation used during the simulations presented later.

```
<object>
  <class>table</class>
  <filter>
    <type>Kalman filter</type>
    <motion_model>zeroth order</motion_model>
    <system_noise>
      <matrix> ... </matrix>
    </system_noise>
    <measurement_noise>
      <matrix> ... </matrix>
    </measurement_noise>
  </filter>
</object>
<object>
  <class>human</class>
  <filter>
    <type>multiple model Kalman filter</type>
    <motion_model>zeroth order, first order<motion_model>
    ...
  </filter>
  <appearance_model>
    <appearance_relation target="door"
        max_distance=0.5 frequency=0.1 />
  </appearance_model>
  ...
<\object>
```

Listing 1. Example XML snippet that can be used to store object specific information

A. Adding knowledge to the MHF

1) Instantiation of a new object: As mentioned in Section III, each time a new object is added to the tree, an object filter is initialized at the observed position. The filter type depends on the dynamic properties of the object, *e.g.*, tables get a Kalman filter with zeroth order motion model whereas humans might need more advanced filters. If there is prior knowledge available about the required filter settings, *e.g.*, obtained from previous runs, it can be added to object models and can then be used to initialize the filter. This knowledge can be represented in any xml-based format, which can then easily be connected to an existing object database as [12]. An example of an xml-based template is given in Listing 1.

2) $p(h_k^i|h_{k-1}^{i'})$ - new objects and appearance models: Updating the probability using (1) involves calculating $p(h_k^i|h_{k-1}^{i'})$. In case of associating the measurement with a new object, $p(h_k^i|h_{k-1}^{i'})$ represents the probability that objects (i) enter the observation region or (ii) reappear after being lost due to occlusion. Especially the first profits from object specific knowledge represented by an appearance model. Typically these appearance models contain common sense knowledge such as 'people enter through the door', spatial relations such as 'computers can be found on a desk' or context dependent information like 'it is not very probable to find a sheep in an office environment'. Open Mind Indoor Common Sense (OMICS) [14] already stores this type of knowledge, *e.g.*, 'coffee is made in a coffee maker which is in a kitchen' and for that reason, it seems a good candidate for providing appearance models. Alternatively, the object relations can be stored in networks [20], *e.g.*, Bayesian networks where nodes represent objects and links the conditional probabilities between the objects.

3) $p(h_k^i|h_{k-1}^{i'})$ - existing objects: This defines the probability that a particular object is updated, *i.e.*, how often does a measurement correspond to this particular object, which

depends on the frequency at which the perception detects that object. Section IV-B will give an example of how this probability can be selected.

4) $p(h_k^i|h_{k-1}^{i'})$ *- clutter:* Characteristics of the detection algorithm or the measurement range of a sensor provide valuable information. For example, if one tries to detect faces using a camera, the percentage of false positives typically depends on tunable parameters in the detection algorithm or the distance between face and camera. Both the percentage and the distance can be included in the probabilistic model for $p(h_k^i|h_{k-1}^{i'})$ in case of clutter. However, for sake of simplicity the distance dependency is omitted for now and will be part of future work.

5) $p(z_k|h_k^i)$ *- existing object:* In case of associating the measurement with an existing object, the likelihood can be determined by the distance between measurement and object, as defined by (2). The measurement noise can be used in a rather straightforward manner using the covariance matrix Σ. In addition, the difference between object and measurement class label can be reflected in the likelihood. Comparing class labels can be done using taxonomic knowledge. If, for example, the taxonomy defines that 'BottleOfCoke' is-a 'Bottle', associating a measurement with class label 'Bottle' with a world model object that has class label 'BottleOfCoke' must have a high likelihood compared to a world model object that has class label 'Chair'. Taxonomies and ontologies can be used for this purpose, *e.g.*, KnowRob [15] can provide such knowledge.

Furthermore, classes can have similar visual appearance, *e.g.*, a mug can contain a picture of a face. In this case, associating a measurement with class label 'Face' with a previously detected 'Mug' can be the best solution, despite the confusing labels. Implementing this requires an advanced ontology, including visual similarity as property between classes, which depends on the detection algorithm used. These ontologies are not the primary focus of this work and for that reason, visual similarity will not yet play a role in this likelihood.

6) Leaving objects: As mentioned in Section III-E, the MHF needs a strategy to deal with leaving objects. Here, it is proposed to use the state uncertainty for this purpose. The state uncertainty increases with every propagation step and typically decreases after an update. The mismatch between observed motion and estimated motion determines the rate at which this uncertainty increases and is dependent on the object. Active objects can move randomly and hence their movements are hard to predict and their uncertainty increases rapidly, *e.g.*, playing children, whereas passive or static objects can be propagated with a smaller increase of uncertainty, *e.g.*, a table or someone watching television. How the uncertainty is propagated depends on the filter, which in turn depends on the object class.

B. IMPLEMENTATION

This section describes how the proposed approach presented in Section IV-A is implemented.

If a new object is added to a hypothesis, a Kalman filter using zeroth and first order motion models is initialized. In order to allow the combination of different filters, an autonomous multiple model (MM) estimator is implemented. More specifically, an autonomous MM estimator with hard B-best approach decisions is used, where $B = 1$ [19]. The combination of a zeroth and first order motion model allows start-stop motions. This autonomous MM filter is used for tracking all moving objects. For objects that are not expected to move, such as tables, Kalman filters with a zeroth order motion model are used. Besides the motion model, initial state covariance and the uncertainty threshold used for dealing with leaving objects are chosen based on the object class. The measurement model depends on the class and detector, *i.e.*, how accurate can the detector detect an object of that class. In the current implementation, all prior information is represented in a file, similar to Listing 1.

Probability updates of the hypotheses are implemented using (1). The knowledge needed for the conditional update probabilities $p(h_k^i|h_{k-1}^{i'})$ is represented using the following three functions:

- $f_{clutter}$: The detection frequency of false positives (clutter), implemented as a constant depending on the detection algorithm and sensor used for detection.
- $f_{existing}(C(o))$: The detection frequency of objects with class $C(o)$, implemented using a look-up table containing detection frequencies per object class.
- $f_{new}(o, h_k^i)$: The frequency of the appearance of object o in context of hypothesis h_k^i, *e.g.*, how often are chairs detected if the objects in h_k^i are all office related. It is implemented as:

$$f_{new}(o, h_k^i) = \max_{o' \in h} f_{presence}(o, o'), \tag{4}$$

  where $f_{presence}(o, o')$ denotes the frequency of the appearance of object o in the presence of o'. As an example, (4) can be used to denote that objects of class 'Person' appear near a door every 10 seconds on average.

These frequecies are designed using the suggestions in Section IV-A, *e.g.*, the frequency at which a face detection module runs or the approximated percentage of false positive based on tunable parameters. Using these functions, the conditional update probabilities $p(h_k^i|h_{k-1}^{i'})$ are calculated as follows:

$$p([h_{k-1}^i, clutter]|h_{k-1}^{i'}) = \frac{f_{clutter}}{F} \tag{5}$$

$$p([h_{k-1}^i, o_{exis}]|h_{k-1}^{i'}) = \frac{f_{existing}(C(o_{exis}))}{F} \tag{6}$$

$$p([h_k^i, o_{new}]|h_{k-1}^{i'}) = \frac{f_{new}(o_{new}, h_{k-1}^{i'})}{F}, \tag{7}$$

where o_{exis} is an existing object that is associated with measurement z_k, o_{new} is a new object added to the hypothesis based on z_k, and the normalizing factor F is the sum of all frequencies.

The likelihood of hypothesis h_k^i is represented as an extension of (2):

$$p(z_k|h_k^i) = L_{class}(C(o_k), C(z_k)) \cdot e^{-\frac{1}{2}\delta^T \Sigma^{-1} \delta}, \quad (8)$$

where $L_{class}(C(o_k), C(z_k))$ denotes the likelihood of object o_k being of class $C(o_k)$, while the measurement is of class $C(z_k)$. If $C(o_k)$ denotes a super- or a subclass of $C(z_k)$, L_{Class} is set to one, otherwise it is set to zero. A simple taxonomy was used to declare such is-a-relations between object classes. The class likelihood function can also be used to declare visual similarities between object classes, but this is future work.

For new objects, (8) always evaluates to one, since the object position and class are chosen based on the measurement z_k. In the case of clutter, the likelihood $p(z_k|h_k^i)$ is chosen to be one.

To sum up, object descriptions used in this paper contain:

- An object class label
- Frequencies of detection ($f_{existing}$), appearance related to other object classes ($f_{presence}$) and false positives ($f_{clutter}$)
- Is-a-relations to other classes, *i.e.*, the object is included in a taxonomy or an ontology
- Filter type and initial conditions, such as initial state covariance, dynamic model, and system noise
- Detector dependent measurement model including measurement noise

V. SIMULATION RESULTS

A. Simulation 1

In the first simulation, the simple taxonomy shown in Fig. 1 was implemented and the Cans and Cups are detected more often. The taxonomy allowed associating detections with class label 'Can' with world model objects with class label 'CokeCan' or 'BeerCan' and vice versa. As a result, the view of the world was consistent and there did not exist multiple instances of the same object with different class labels. Furthermore, measurements with other class labels, *e.g.*, table or cup, were only associated with objects that had the same class label, or with clutter. Detections with a wrong class label, typically were associated with clutter.

The tracking filtered the noisy detections, which resulted in a steady world model. No dynamics were involved, hence the Kalman filters had zeroth order motion models.

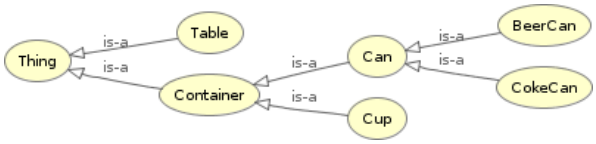

Fig. 1. Taxonomy used during simulation 1.

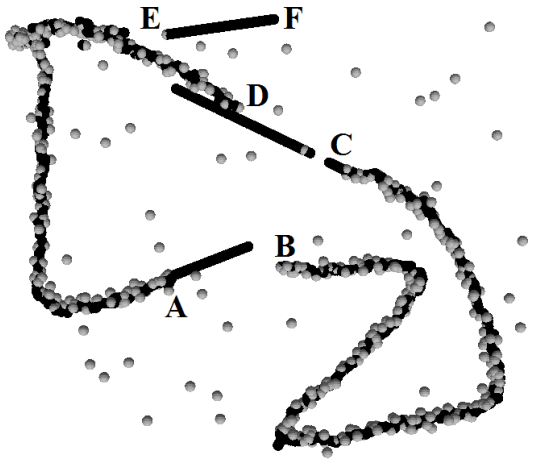

Fig. 2. Simulated measurements (gray dots) including false positives, and output of the MHF (black dots). Between section (A, B) and (C, D) occlusion was simulated. (E) corresponds to an incorrect object instantiation, which was propagated to and removed at (F).

B. Simulation 2

In this simulation, the task is to track a person. Fig. 2 shows the setting and the results of the second simulation. In the upper left corner, a door was assumed. Based on simple knowledge, *i.e.*, 'people enter through a door', the frequency f_{new} of a new object appearing close to the door is chosen to be higher than the frequency elsewhere. The simulated person enters and leaves through the door. The position measurements are noisy and false positives appear randomly over the whole observation area, as indicated by the gray circles in Fig. 2. The person walks anti-clockwise and all measurements over the whole time interval are shown. Both between points A and B and between C and D, a 1.5 [s] occlusion is assumed and for that reason, only false detections are available in these time intervals. The black dots show the trajectories of objects according to the world model.

Up till point A, the world model generates the result that is desired, *i.e.*, a smooth trajectory through the noisy detections. Once the person is occluded, the position is propagated using the Kalman filter with first order motion model, resulting in the black line. At point B the person re-appears, but due to the nonlinear trajectory of the person during the occlusion the propagation was slightly off. However, once the person re-appears at point B, the tracking enables associating with the propagated person and the mismatch between propagation and measurement is eliminated.

From point B to C, everything again goes as expected. Then the person is occluded again. Due to a coincidental false detection on the left of point C, the estimated velocity increases. As a result, the propagated position and the measurements after re-appearance again are different. Still, the tracker can catch up with the person at point D and keeps

track until the person leaves the room.

Note that there is one incorrect object trajectory between E and F. Two false detections appear close to each other and within a short time interval near point E. As a result, the world model assumes a person, despite the relatively low f_{new} in this region. After the update with the second false positive, the person is not detected anymore and a propagation is performed. During this propagation the uncertainty on the person's position increases. At point F, the uncertainty is above the threshold and the object is removed from the world model.

VI. CONCLUSIONS AND FUTURE WORK

A. Conclusions

This paper presented methods for adding object knowledge to a world model, such that it performs better in unstructured, dynamically changing environments under the constraint of real-time applicability. The world model was implemented using a Multiple Hypothesis Filter, which is a structure that aims at solving the data association problem. The probabilistic models underlying the MHF are crucial for the quality of its output. It was shown how knowledge can be used to improve those models:

- Use spatial relations between objects to determine the probability of the appearance of new objects
- Choose the appropriate tracking filter depending on the dynamic properties of the object
- Use detector-specific knowledge, such as frequency of detection and number of false positives to determine the reliability of a measurement
- Use taxonomic knowledge about object classes to enable a more flexible association between objects and measurements

Furthermore, this paper proposed an uncertainty-based strategy for the active removal of leaving objects. Uncertainty propagation is based on the filter used for tracking. By using multiple model filters and object class dependent filter settings, the removal of leaving objects now depends on the knowledge about the dynamic properties of the objects.

The results of simulations confirmed the potential of incorporating object knowledge in a MHF.

B. Future work

First of all, the future work will be:

- Show the benefits of incorporating object knowledge during real-life experiments
- Using more general knowledge, *e.g.*, [13]–[14]

Besides this, extending the probabilistic models, using more advanced multiple model tracking strategies [19] and using visual similarity between objects to calculate the likelihood of association with existing objects can further improve the results. Finally, it is aimed to use RoboEarth [21] to share the knowledge required and obtained by the algorithm described in this paper.

VII. ACKNOWLEDGMENTS

The research leading to these results has received funding from the European Union Seventh Framework Program FP7/2007-2013 under grant agreement n°248942 RoboEarth.

REFERENCES

[1] Y. Bar-Shalom and T. Fortmann, *Tracking and Data Association*, ser. Mathematics in Science and Engineering. Academic Press, 1988.
[2] D. Reid, "An algorithm for tracking multiple targets", *IEEE Trans. on Automatic Control*, vol AC-24, no. 6, pp 843–854, December 1979.
[3] Ronald P.S. Mahler, "Multitarget Bayes Filtering via First-Order Multitarget Moments", *IEEE Trans. on Aerospace and Electronic Systems*, vol 39, no. 4, pp 1152–1178, 2003.
[4] T. De Laet, "Rigorously Bayesian Multitarget Tracking and Localization", Ph.D. dissertattion, Katholieke Universiteit Leuven, 2010.
[5] J. Elfring, M.J.G. van de Molengraft, R.J.M. Janssen and M. Steinbuch, "Two Level World Modeling for Cooperating Robots Using a Multiple Hypotheses Filter", *Int. Conf. on Robotics and Automation*, Shanghai, 2011.
[6] Nico Blodow, Dominik Jain, Zoltan-Csaba Marton and Michael Beetz, "Perception and Probabilistic Anchoring for Dynamic World State Logging", *Proc. of 2010 IEEE-RAS Int. Conf. on Humanoid Robots*, Nashville, TN, USA, 2010.
[7] `http://i61p109.ira.uka.de/ObjectModelsWebUI/`.
[8] `http://vision.caltech.edu/Image_Datasets/Caltech256/`.
[9] `http://www.behold.cc/`.
[10] `http://marathon.csee.usf.edu/range/DataBase.html`.
[11] `http://sketchup.google.com/3dwarehouse/`.
[12] `http://www.ros.org/wiki/household%20objects`.
[13] C. Matuszek, J. Cabral, M. Witbrock, and J. DeOliveira, "An introduction to the syntax and content of Cyc", *Proc. of the 2006 AAAI Spring Symposium on Formalizing and Compiling Background Knowledge and Its Applications to Knowledge Representation and Question Answering*, Minneapolis, pages 44–49, 2006
[14] Rakesh Gupta and Mykel J. Kochenderfer, "Common sense data acquisition for indoor mobile robots", *Nineteenth National Conference on Artificial Intelligence*, pages 605–610, 2004
[15] Moritz Tenorth and Michael Beetz, "KnowRob Knowledge Processing for Autonomous Personal Robots", *IEEE/RSJ Int. Conf. on Intelligent RObots and Systems*, 2009
[16] Amy Loutfi, Silvia Coradeschi, Marios Daoutis, and Jonas Melchert, "Using Knowledge Representation for Perceptual Anchoring in a Robotic System", *Int. Journal on Artificial Intelligence Tools*, 17(5), pages 925–944, 2008
[17] Stefano Coraluppi, Craig Carthel, Peter Willett, Maxence Dingboe, Owen O'Neill and Tod Luginbuhl, "The Track Repulsion Effect in Automatic Tracking", *12th Int. Conf. on Information Fusion*, Seattle, WA, USA, July 2009.
[18] M. Betke, D.E. Hirsh, A.Bagchi, N.I. Hristov, N.C. Makris, and T.H. Kunz, "Tracking Large Variable Number of Objects in Clutter", *IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*, Minneapolis, MN, June 2007
[19] X. Rong Li and Vesselin P. Jilkov, "Survey of Maneuvering Target Tracking. Part V: Multiple-Model Methods", *IEEE Trans. on Aerospace and Electronic Systems*, vol 41, no 4, October 2005
[20] F.V. Jensen and T.D. Nielsen, "Bayesian Networks and Decision Graphs" Second edition, Springer, 2007
[21] O. Zweigle, M.J.G. van de Molengraft, R. d'Andrea and K. Häussermann, RoboEarth: connecting robots worldwide, *in ICIS '09: Proc. of the 2nd Int. Conf. on Interaction Sciences*, 2009, pp 184–191.