

Multi-Resolution particle filter estimation, applied to object recognition in an office environment

Tinne De Laet, Wim Meeussen, Joris De Schutter and Herman Bruyninckx

Abstract—This paper presents a multi-resolution particle filter, applied to the recognition and localization of objects in an office environment, using a laser scanner. A fast low level particle filter combines multiple measurements of the Sick laser scanner to localize the legs of tables and chairs in an office environment. This low level filter uses a simple leg model to distinguish the legs from other objects in the office environment. A high level particle filter then combines the estimated leg positions to recognize tables and chairs, and find their position and orientation. The high level filter is helped by the knowledge of the geometry of the tables and chairs. This geometric knowledge is modeled using a simple Bayesian network. The estimation of the high level filter is used to correct the low level estimation, to add missing legs that were not found by the low level filter, and to remove legs that are not part of a table or a chair. The combination of the fast low level tracking with the high level geometric knowledge, increases the performance of the estimation problem, and allows for realtime object recognition and tracking in a 50 [m²] office environment. Experimental results show the effective recognition of two tables and two chairs in an unstructured office environment.

Index Terms—multi-resolution, particle filter, object recognition

I. INTRODUCTION

Many problems in robotics require the estimation of a high dimensional unknown state, based on sensor measurements that provide indirect and noisy information about this state. In the Bayesian approach, the *probability density function* (pdf) that represents the unknown state, is updated recursively with the available sensor measurements. Each sensor measurement is linked to the unknown state with a measurement model. The evolution of the state is described by a system model. For many applications in robotics research, the Bayesian sequential Monte Carlo method, or particle filter [1], [3], is used for non-linear estimation problems. A particle filter approximates the state pdf with a set of discrete particles. This discrete representation allows a particle filter to work with multi-modal state pdf's of any shape, however, at the expense of a high number of particles required. The number of particles required, increases drastically for each extra dimension in the unknown state. Many applications use thousands of particles to represent discrete state pdf. In a particle filter, the discrete pdf is updated with the measurement information by applying the measurement model

All authors are with the Department of Mechanical Engineering, Katholieke Universiteit Leuven, Belgium.

Corresponding author: Tinne De Laet
(tinne.delaet@mech.kuleuven.be)

All authors gratefully acknowledge the financial support by K.U.Leuven's Concerted Research Action GOA/05/10

Tinne De Laet is a Doctoral Fellow of the Fund for Scientific Research-Flanders (F.W.O.) in Belgium

to each of the particles. These many measurement updates make particle filters computationally expensive. For realtime state estimation, this computational cost often prevents the use of particle filters.

To reduce the number of particles required, and hence reduce the computational cost, it is important to use an accurate prediction model for the unknown state, such as for example in [8]. This is of course application-specific and is not always possible. Therefore, many application-independent approaches have been proposed to reduce the computational cost of a particle filter. Evolutionary approaches apply genetic algorithms such as crossover and mutation operations to a particle distribution; this reduces the number of particles required in the particle filter, and therefore decreases the computational cost of the filter [6], [9]. Doucet et al. [2] show how the structure of a Dynamic Bayesian Network can be used to increase the efficiency of particle filtering by using a technique known as Rao-Blackwellisation. Essentially, this technique samples some of the variables and marginalizes out the rest of them exactly with any finite dimensional optimal filter such as a Kalman filter. A variable resolution particle filter is proposed in [12], to reduce the number of particles required. The approach dynamically changes the particle resolution, to a fine resolution where the belief is strong, and a coarse resolution where the belief is low, while preventing potential hypotheses from being eliminated.



Fig. 1. The multi-resolution particle filter is applied to localize and recognize objects in an unstructured office environment.

In this paper we propose a multi-resolution particle filter, where the estimation process is divided into a fast *low level* filter with a low-dimensional state model, and a *high level* filter with a higher-dimensional state model. For example,

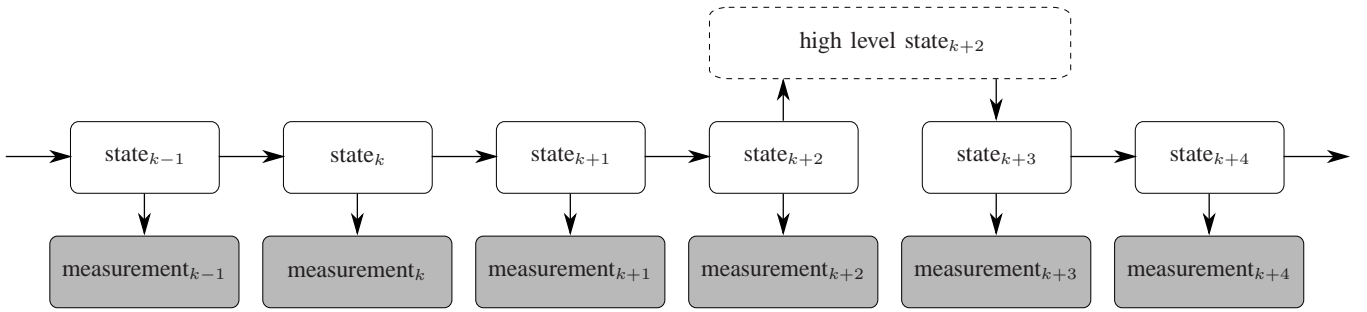


Fig. 2. The causal relation between the high level state, the low level state and the sensor measurements is expressed in a Bayesian network. The grey nodes denote the sensor measurements, the full transparent nodes denote the unknown positions of the table legs and chair legs, and the dashed transparent nodes denote the unknown position and orientation of the tables and chairs.

the low level filter can estimate the edges of a cube (4 DOF per edge) based on camera measurements, and the high level model combines these edges to estimate the complete cube (6 DOF pose). A similar example is discussed in this paper, where the low level filter estimates the legs of tables and chairs in an office environment, and the high level filter combines the legs estimates into tables and chairs, as shown in Fig. 1. In general, the low level filter is updated with all available sensor information at each time step. The high level filter is updated at a much lower rate, with the state estimation of the low level filter. This system reduces the amount of data that is processed by the computationally expensive high level filter.

The paper is organized as follows. Section II presents the multi-resolution particle filter, with its low level and high level filter. In Section III the multi-resolution particle filter is applied to estimate the position and orientation of tables and chairs in an office environment. The real world experimental results in an office environment are presented in Section IV. Finally, Section V contains the conclusions and future work.

II. MULTI-RESOLUTION PARTICLE FILTER

A classical particle filter approach directly updates its (often high dimensional) unknown state with all available sensor data, using a fully detailed model. This is computationally expensive and therefore often not possible in realtime. The proposed multi-resolution particle filter in this paper aims at reducing the computational cost of an estimation problem, by using a multi-step approach with multiple filters. In this paper a two-step approach is discussed, with a low level filter and a high level filter. The low level filter uses a simplified, low dimensional model, and is updated in each time step, with all available sensor data. The fully detailed model in the high level filter is not applied in every time step, but only at a much lower rate. In its update cycle, the high level filter is first updated with the estimate of the low level filter, and then the low level filter is corrected with the high level estimate. This procedure drastically reduces the amount of data processed by the more detailed and computationally more expensive high level filter.

A. Low level filter

Like in the classical particle filter approach, the low level filter is updated with all the available sensor measurements. However, the low level filter only uses a partial or simplified model. Compared to a classical particle filter, the overall computational cost of the multi-resolution particle filter is lower, if:

- the unknown state of the low level filter is lower dimensional than the state of the classical particle filter, and therefore requires significantly less particles, and
- the low level filter uses a partial or simplified model, that is more easily linked to the available sensor measurements than a complete model.

The application in Section III gives an example of a low dimensional state that fulfills these conditions.

The low level filter is a classical particle filter, as represented by the Bayesian network [5] in Fig. 3. This network is updated recursively in a two step approach. In the first step a system model predicts the state of the system in the next time step. In the second step a measurement model corrects this prediction based on the sensor measurements.

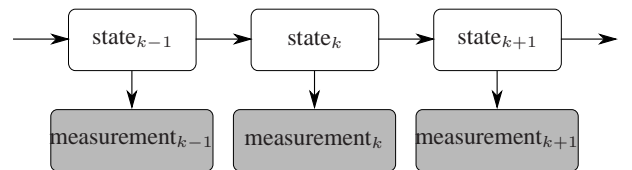


Fig. 3. The causal relation between the low level state and the sensor measurements is expressed in a Bayesian network. The grey nodes denote the sensor measurements, while the transparent nodes denote the unknown low level state. A particle filter is used to update this network recursively.

B. High level filter

Unlike the low level filter, the high level filter contains a complete model, and therefore computational cost of the high level filter is often high. However, the high level filter is not updated with every single sensor measurement; it is only updated at a much lower rate, with the estimation results of the low level filter. Because the high level filter contains a complete model, its estimate is more complete and accurate

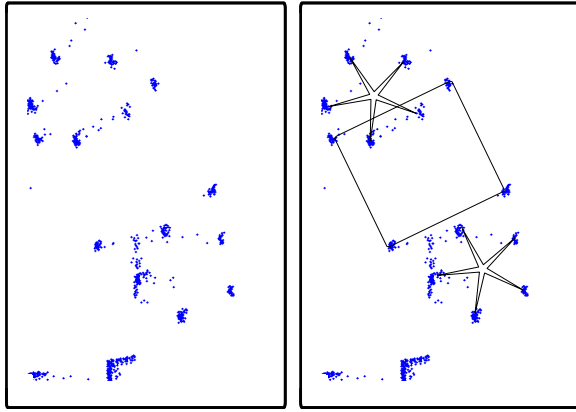


Fig. 4. The laser scanner measures the distance the legs of tables and chairs. The left side shows a small part of a laser scan where some of the legs of tables and chairs are visible. The right side shows the same part of the laser scan, with the actual positions and orientations of the four-legged table and five-legged chairs marked.

than the estimate of the low level filter. Therefore, after the high level filter is updated, the information in the high level is used to correct and complete the low level estimate.

The Bayesian network that includes both the low level and the high level estimation problem, is shown in Fig. 2. It shows how from time step $k - 1$ to $k + 2$ only the low level filter is applied to update the low level estimate, based on the laser scan measurements. Between time step $k + 2$ and $k + 3$ the information contained in the low level estimate is first used to update the high level estimate. Then, the high level estimate is used to create a new and corrected low level estimate. Starting from this new and corrected low level estimate the low level filter continues for a sequence of updates based on the laser scan updates.

III. OFFICE ENVIRONMENT APPLICATION

In this section, the multi-resolution particle filter approach is applied to the recognition and localization of objects in an office environment. In this application, a mobile robot navigates in an office environment, while a Sick laser scanner observes the environment. The laser scanner measures the distance to objects in the environment; the laser scanner measures the distance to the legs of tables and chairs. Fig. 4 shows a small part of a laser scan where some of the legs of a table and two chairs are visible, expressed in Cartesian space. This figure illustrates it is not a trivial task to recognize tables and chairs from the raw laser scan data, even for this small area. The data in the left figure appear to have no structure, and without the geometric knowledge of the tables and chairs many groups of legs could be combined into a table or chair.

In a first step, a low level filter estimates the positions of the legs, from the laser scanner data. In the second step, the position and orientation of the tables and chairs is estimated from the estimated leg positions. The same low level based on a leg model, and high level filter based on the complete object model, could be used for other applications, such as pallet recognition in an industrial environment [7], [11].

A. Low level leg estimation

The low level particle filter only models the legs of a table or a chair, but does not contain the information that a table has four legs or chair has five legs, in a specific configuration. While the complete object model is a hybrid 4-dimensional model (continuous x-y positions and orientation, and discrete choice between table or chair), this simplified model is only 2-dimensional (x-y positions). Hence the number of particles required—and therefore also the computational power required—is reduced drastically¹. On top of that, the computational cost of the measurement model at the low level filter is lower, thanks to the simplicity of a leg model. This is important, because a single laser scan contains 360 point measurements, and the measurement model links each of these point measurements to each of the particles. The measurement model consists of a number of zones. For a possible leg position, multiple zones around the center of the leg are distinguished, as shown in Fig. 5. Each point measurement of the laser scan influences the belief in a possible leg position in a different way, depending on the zone it lies in. A point measurement in zone 1 indicates that the front of the leg was seen by the laser scanner, and therefore this measurement increases the belief in the leg position. Zone 2 is the empty space that is assumed around each leg position. A point measurement in this zone decreases the belief in the leg position. A point measurement in zone 3 would mean that the laser beam went straight through the assumed leg position, which is physically impossible; a point measurement in this zone therefore also decreases the belief in the leg position. A point measurement in zone 4 indicates that an obstacle in between the laser scanner and the assumed leg position blocks the view; therefore such a point measurement provides no information about the assumed leg position. A point measurement in zone 5 provides no information about the assumed leg position. Unlike what Fig. 5 suggests, in reality the transitions between the zones are smoothed.

This simple zone based measurement model is able to:

- distinguish point measurements on the leg of a table of chair from point measurements of for example a wall. When the laser scanner sees a wall, many point measurements lie next to each other on a long line; there will always be point measurements that lie in zone 2, and therefore a wall or a large object will be discarded by the low level filter.
- estimate the position of round legs, as well as the position of differently shaped legs such as for example square legs, as long as the shape can be fitted inside zone 1.
- take into account the complete path of the laser beam, and not only the end point where the beam touches an object. When the path of a laser passes through a

¹when for example the average distance between the particles for the orientation is chosen to be 2° , the number of particles required in the low level filter is reduced by a factor 180. In addition, the legs of a table and a chair are the same, making the low level filter 360 times faster than a full dimensional filter.

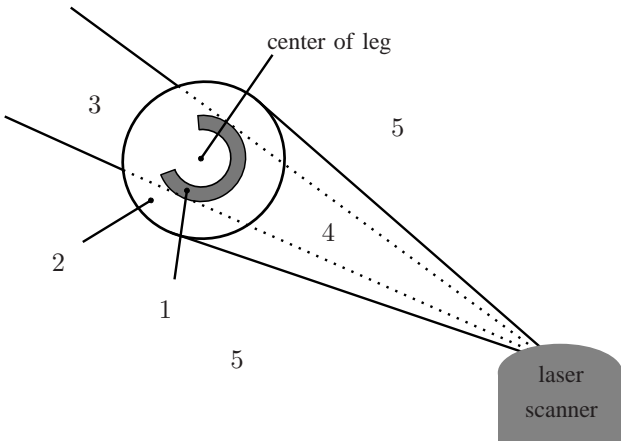


Fig. 5. The zones that are considered in the low level leg template: a measurement in zone 1 increases the belief in the leg, a measurement in zone 2 or zone 3 decreases the belief in the leg, and a measurement in zone 4 or zone 5 gives no information about the leg.

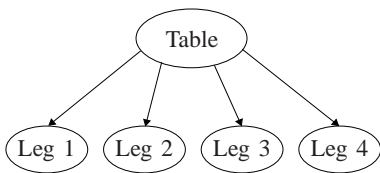


Fig. 6. Simple Bayesian network containing the geometric knowledge of a table.

possible leg position, the belief in this leg position is decreased.

- recognize when a laser scan measurement is not informative about a possible leg position (zone 4 and 5). In this case the belief in the leg position remains unchanged.

B. High level object recognition

The high level filter uses a complete geometric model of the tables and the chairs. This complete model is 4-dimensional hybrid model: 3-dimensional continuous (x - y positions and orientation), and 1-dimensional discrete (table or chair). The complete model describes how a table has four legs and a chair has five legs, in a specific configuration. This complete geometric model of the objects at the high level filter is modeled using a simple Bayesian network. An example network for the table is shown in Fig. 6. The nodes represent a table and its four legs, and the arrows between the nodes contain the table-specific geometric information. For the chair a similar network is used.

The high level estimate is updated with the estimated leg positions of the low level filter (soft evidence [5]). The measurement model describes, for a position and orientation of a table or chair, what the expected leg positions are. While the low level leg estimate is updated in every time step, the high level filter is only updated at a much lower rate. The high level filter provides the information that is searched for in this application: the probability to find a table or a chair

at a certain position and orientation.

The high level is also able to “correct” the low level estimate. When for example three of the four legs of a table were recognized in the low level estimate, but not the fourth leg, the high level filter will still recognize this as a table. Because the high level has the knowledge of the geometry of a complete table, it also knows where the fourth leg should be found. This information about the fourth leg is inserted back into the low level filter. This shows how the high level filter can insert new information into the low level filter. Also, when the low level filter finds a leg that is not part of a table or a chair, the high level filter will identify this leg as “not part of a table or chair”, and remove it from the low level estimate. This shows how the high level filter can remove useless information from the low level filter. The result is that, from time to time, the low level filter is corrected based on the complete geometric knowledge of the objects in the environment.

IV. EXPERIMENTS

The presented multi-resolution particle filter approach has been implemented and verified in a real world experiment. The implementation is based on the *Bayesian Filtering Library* (BFL) [4]. The experimental setup consists of a $50 m^2$ office environment ($10 m \times 5 m$), as shown in Fig. 1. During a period of 10 *sec* the laser scanner moves about 1 *m* in its environment, while it takes measurements of the environment, at a rate of 1 *Hz*. Each measurement contains 360 point measurements, over an angle of 180° , at 0.5° increments, with a maximum range of 8 *m*. Fig. 7 combines all 360 measured points of each of the 10 measurements taken by the laser scanner in this experiment. The scans give an indication of the leg position of two tables and two chairs around coordinates (2.5 *m*, 1 *m*), and the position of the walls at the border of the scan. The scan shows that the experiment is performed in an unstructured environment, since the scan detects many other objects besides the table and the chairs.

A. Low level particle filter

In the $50 m^2$ environment, the low level particle filter uses 50,000 particles to estimate the unknown leg positions. Each particle represents a 2-dimensional x - y position of a leg. The average distance between each two particles is 3.0 *cm*. Because no a priori information about the location of the table and the chairs is available, at the first time step, the low level filter is initialized with a uniform particle distribution over the whole environment. At each time step, each of the particles is updated with all 360 measured points. When operating in realtime on a 2 *GHz* AMD64 laptop, the low level filter only uses 50 % of the available computation time. This leaves time for the processing of the high level filter, which is not realtime and takes longer than one time step.

Fig. 8 shows the posterior state pdf of the low level filter, represented by the particle distribution, after 10 complete laser scans are processed. The colors indicate the probability of a leg at the x - y position. The legs that are recognized by

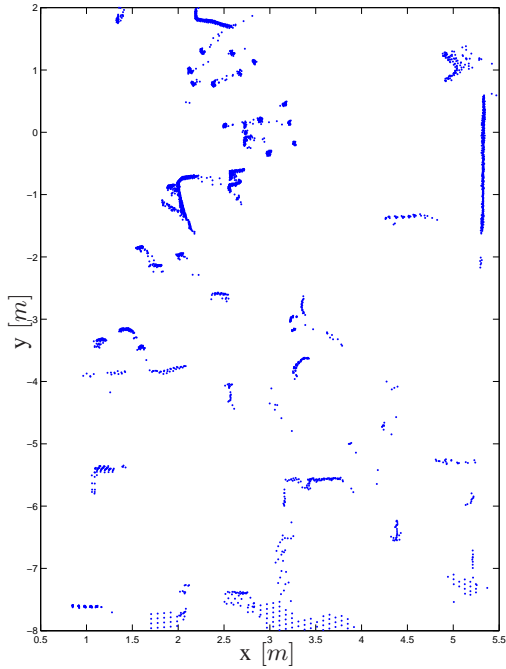


Fig. 7. The laser scanner measures the distance to all obstacles in a 50 m^2 office environment. Each scan consists of 360 point measurements, over a range of 180° , at 0.5° increments.

the low level filter are marked as highly probable by the dark red color. The filter finds most legs of the table and chairs in the area around $(2.5 \text{ m}, 1 \text{ m})$, as well as some other objects in the environment with a similar shape and size. The low level filter cannot distinguish between the legs of the table and chairs because it only uses a partial model. The dark blue color indicates the areas with a low probability; possible leg positions in this area where “penetrated” by a laser beam, indicating that no real leg is present at these positions, or larger objects like a wall. The lighter blue color marks the areas with a medium probability; the laser beam never reached these areas, and therefore there is no knowledge about possible legs. In these light blue areas the particle density does not drop after re-sampling the discrete pdf, and therefore the particle filter will still be able to recognize the legs in this area when a new laser scan provides information about the area.

B. High level particle filter

The high level filter recognizes the tables and chairs in the environment, and estimates their x-y position and orientation. While the low level filter is updated in every time step, the high level filter is only updated every 10 time steps. Instead of initializing the 4-dimensional hybrid state pdf (3-dimensional continuous and 1-dimensional discrete) with a large uniform particle distribution, the prior pdf of the high level filter is initialized with the leg estimation of the low level filter; hence no particles are “wasted” in areas where

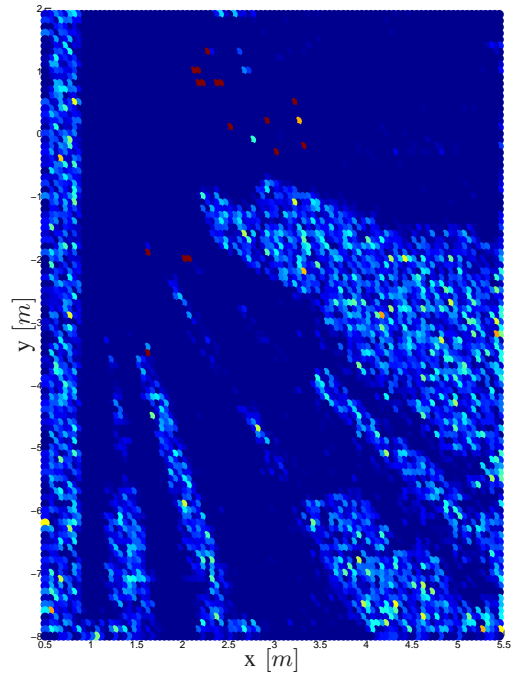


Fig. 8. The posterior state pdf estimated by the low level filter. The colors indicate the probability of a leg at a position. The red color indicates a high probability, the dark blue color indicates a low probability, while the light blue color indicates that no information about the possible leg position is available.

no legs are found. For the initialization, the low level leg estimation is represented as a histogram, and the 50 most probable legs are extracted, as shown by the dots in Fig. 9. At each of these 50 leg positions, the leg of a table and chair is attached, and then rotated around 360° in 100 steps of 3.6° . This results in an initial distribution with 50×100 possible table positions/orientations and 50×100 possible chair positions/orientations.

After the initialization, the high level state pdf is updated with the 50 most probable leg positions and their probability, as estimated by the low level filter (soft evidence [5]). The high level filter uses the *Bayesian Net Toolbox* (BNT) [10]. Fig. 9 shows the 50 most probable leg positions provided by the low level filter as dots, and the most probable table and chair positions/orientations are indicated by their outline. After the high level filter is updated, the low level filter is corrected with the estimate of the high level filter. This is illustrated in Fig. 10, where the left side shows the original low level estimate, and the right side shows the corrected low level estimate. The corrected low level estimation is then updated with the next 10 laser scans, followed by another update of the high level filter, etc.

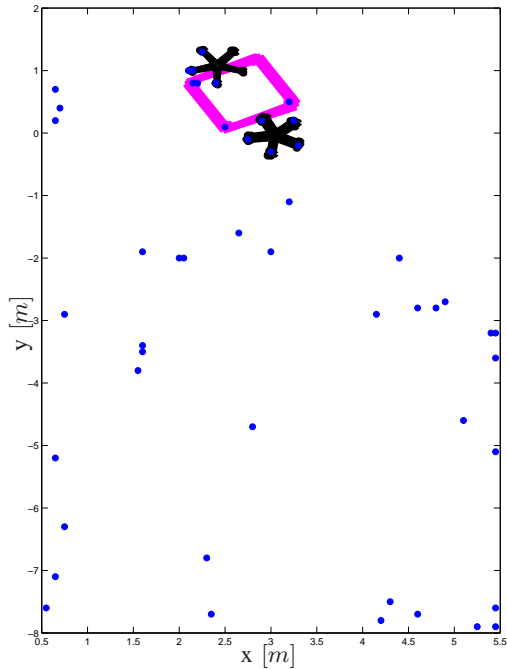


Fig. 9. The high level filter recognizes the position and orientation of the tables and chairs in the office environment. The objects in this figure are reshaped because the proportions were not maintained.

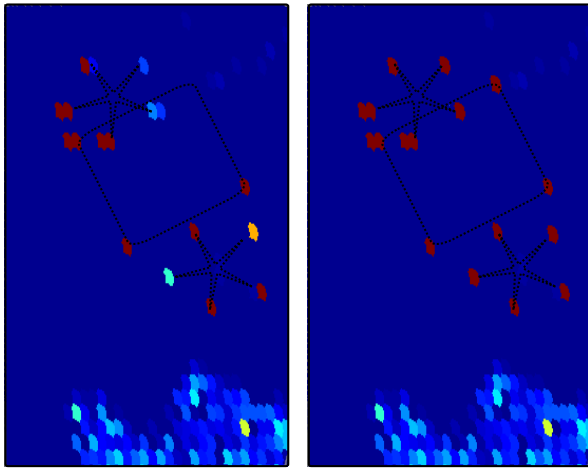


Fig. 10. The low level filter is corrected by the estimation of the high level filter. The low level filter did not see all of the legs of the table and the chairs (left figure). The high level filter knows how many legs make a table or chair, and therefore it can add the missing legs (right figure).

V. CONCLUSIONS AND FUTURE WORK

A. Conclusions

This paper presents a multi-resolution particle filter approach, to increase the performance of an estimation problem. Instead of processing large amounts of sensor data with one single filter with a complete and detailed model, the presented approach splits up the estimation problem

in multiple levels; two levels in the presented example: a low level filter and a high level filter. The low level filter only uses a partial model which is low dimensional and computationally inexpensive, while the high level filter uses a complete model. The low level filter processes all available sensor data, and is updated at each time step. The high level filter is only updated at a much lower rate, with the estimation results of the low level filter. After the high level filter is updated, its estimation is used to correct the low level estimation. In the presented example, the computational cost was reduced about 300 times compared to a normal particle filter.

B. Future work

The current implementation of the high level filter is based on Matlab code. To integrate the whole multi-resolution filter in a decision making network on a robot, the high level filter will be implemented in C++. This will also make it possible to integrate specific active sensing actions in each level of the multi-resolution filter. Active sensing actions in the low level aim at improving the laser scan by for example adjusting the angle of the scanner. Active sensing actions on the high level plan larger motions of the robot to gather more information about specific objects in the environment; for example when the low level recognizes three legs, high level active sensing steers the robot towards the position where the fourth leg is expected to be found. The information in the estimation is not only useful for active sensing, but also to replace the time-based triggering of the high level filter with an information-based triggering.

REFERENCES

- [1] A. Doucet, N. de Freitas, and N. Gordon, editors. *Sequential Monte Carlo Methods in Practice*. Statistics for engineering and information science. Springer, january 2001.
- [2] A. Doucet, N. de Freitas, K. Murphy, and S. Russell. Rao-Blackwellised particle filtering for dynamic Bayesian networks. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 176–183, 2000.
- [3] A. Doucet, N. J. Gordon, and V. Krishnamurthy. Particle Filters for State Estimation of Jump Markov Linear Systems. *IEEE Transactions on Signal Processing*, 49(3):613–624, march 2001.
- [4] K. Gadeyne. BFL: Bayesian Filtering Library. <http://www.orocos.org/bfl>, 2001.
- [5] F. V. Jensen. *Bayesian Networks and Decision Graphs*. Springer-Verlag, 2001.
- [6] N. Kwok, F. Gu, and W. Zhou. Evolutionary particle filter: re-sampling from the genetic algorithm perspective. In *Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2935–2940, Edmonton, Canada, 2005.
- [7] D. Lecking, O. Wulf, and B. Wagner. The variable pallet pick-up and 3D localisation in industrial environments. In *European Robotics Symposium*, Palermo, Italy, March 2006.
- [8] W. Meeussen, J. Rutgeerts, K. Gadeyne, H. Bruyninckx, and J. De Schutter. Contact state segmentation using particle filters for programming by human demonstration in compliant motion tasks. *IEEE Transactions on Robotics*, 2006. in press.
- [9] L. Moreno, S. Garido, and M. Munoz. Evolutionary filter for robust mobile robot global localization. *RAS*, 54(7):590–600, 2006.
- [10] R. R. Murphy. The bayes net toolbox for MATLAB. *Computing Science and Statistics*, 33, 2001.
- [11] M. Seelinger and J.-D. Yoder. Automatic visual guidance of a forklift engaging a pallet. *RAS*, 54(12):1026–1038, 2006.
- [12] V. Verma, S. Thrun, and R. Simmons. Variable resolution particle filter. In *International Conference on Artificial Intelligence*, 2003.