Pose Extraction from Sample Sets in Robot Self-Localization – A Comparison and a Novel Approach

Tim Laue, Thomas Röfer
Deutsches Forschungszentrum für Künstliche Intelligenz, Safe and Secure Cognitive Systems, Bremen, Germany

***Abstract*— For state estimation in robotics applications, especially for robot self-localization, the Monte-Carlo approach has been a popular choice in recent years. Since the original proposal of the approach, several modifications and improvements, e.g., for more intelligent resampling or for efficiently coping with the *kidnapped robot problem*, have been proposed. Nevertheless, the currently used approaches for computing a final result given the sample set bear several drawbacks. In this paper, we compare the most common techniques and propose a new approach that is computationally inexpensive and able to deal with multimodal distributions in self-localization scenarios. All results have been gained in experiments with a humanoid robot in a robot soccer scenario.**

***Index Terms*— Self-Localization, Monte-Carlo, Pose Extraction**

I. Introduction

The Monte-Carlo localization approach [2] for robot self-localization has been quite popular for several years. It approximates a probability distribution by a set of discrete samples, also referred to as particles, and is able to represent multimodal distributions and to deal with non-linear motion models. Having originally been applied to driving robots equipped with comparably precise laser range finders, it has found its way to a variety of different robots. Among these are, e. g., walking, vision-based soccer robots with a rather limited field of view and imprecise odometry that have become enabled to localize precisely within a standardized soccer environment [8, 9].

In the RoboCup scenario (and probably in several others), *robot kidnapping* is a common problem. This can be caused by falling down, including undetectable rotations, or by being replaced by a human. By adding new samples to the probability distributions computed from recent measurements, the so-called *sensor resetting* by [7], the Monte-Carlo approach is able to recover quickly from kidnapping actions.

In general, robot behavior and robot control components that rely on an estimate of the robot's pose within a given frame of reference do not deal with probability distributions. Instead, they usually require a single definite pose (often in 2-D) as a base for further computations. This single pose needs to be extracted from the probability distribution and should be the position (in state space) having the highest probability density.

Having a unimodal probability distribution, this is an almost trivial task. However, in many self-localization scenarios, especially when recovering from kidnapping or determining an initial pose estimate (i. e. starting from an unknown position), the probability distributions representing the current pose estimate are multimodal. For this computation different approaches are known, but they have different drawbacks, e. g. computational complexity or precision.

The contribution of this paper is a novel approach for computing a pose given a sample-based probability distribution. The algorithm – labeled *particle history clustering* – is precise, very efficient, and straightforward. This is achieved through keeping the resampling history of each particle from the moment of sensor resetting. The approach has been evaluated and compared to other approaches in real-robot experiments that have been carried out on a humanoid robot in a soccer scenario.

This paper is organized as follows: Section II describes common approaches for computing a pose from a particle set. In Sect. III, the proposed new approach is presented. All methods are compared by experiments described in Sect. IV. The paper does not contain a detailed description of the Monte Carlo localization approach, a detailed introduction is given in [11].

II. Common Approaches

For computing a robot pose given a sample set, several approved approaches already exist. In this section, the most important ones are described shortly together with their specific advantages and disadvantages.

A. Overall Averaging

When having a scenario in which the probability distribution can be assumed to be always unimodal, e. g., when having a known start position and no robot kidnapping, computing the average (whether weighted or not) over all samples is a reasonable choice. Nevertheless, this method completely fails for multimodal probability distributions. Additionally, single outliers might strongly influence the result.

B. Best Particle

A simplistic but the computationally least expensive approach is to select the particle having the best weighting. If a resampling step is performed in each execution cycle (and thus a particle's weight is not kept over time), the best particle depends only on the last measurement. Such a measurement

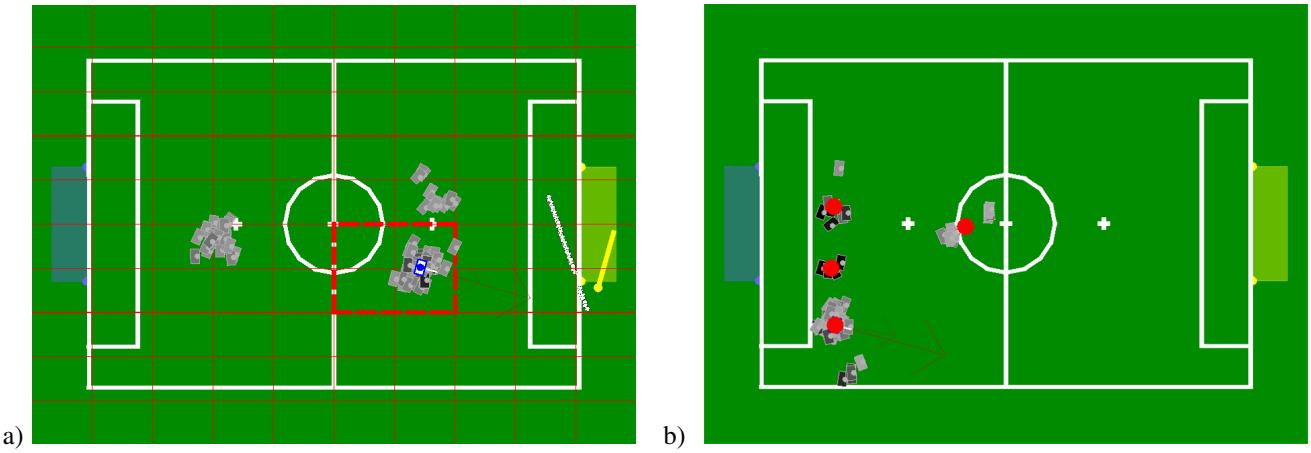

Fig. 1. Visualization of two pose extraction approaches applied to a RoboCup scenario: a) Binning. The environment is divided into 10×10 cells. The 2×2 sub-square containing the most samples is highlighted. b) K-means clustering. Four clusters have been found. Each cluster center is marked by a filled circle.

might be fitting well locally but not globally, e. g., when perceiving a line on a RoboCup field the distance to that line might by perfect according to the current state of a particle, but it might actually be the wrong line.

C. Binning

A computationally inexpensive approach to deal with multimodal distributions is *Binning* [11] discretized the state space into so-called *bins*, as depicted in Fig. 1a. For density extraction, for each bin, the included samples are counted. Within the bin which contains the most samples, an average is computed. For extracting multiple hypotheses, the averaging can also be done for further bins.

Though being a straightforward, robust extension of the general averaging approach, this method has two inherent drawbacks: At first, the number and size of the bins needs to be carefully chosen by the user. Choosing too large bins leads to an improper handling of multimodal distributions, choosing too small bins leads to an instable selection of the best bin. In addition, there always exist instabilities at the edges of the bins. As depicted in Fig. 1a, clusters might be situated at the edge of two bins and thus the extracted robot pose might oscillate between these two bins.

Both problems can be compensated by not only selecting a single bin but a $n \times n$ subspace containing the most samples, this has e. g. been done by [9, 5]. Larger sets of neighboring bins stabilize the output, but the effects of the discretization of space are still present. In addition, they are in conflict with the goal of be able to represent multimodal distributions.

D. K-means Clustering

An obvious continuous solution for extracting multiple robot pose hypotheses from a sample set would be the application of a clustering algorithm, e. g. *k-means clustering* as proposed in [11]. Given a sample set and a maximum number k of possible clusters, the algorithm iteratively converges towards the centers of k sample clusters, as depicted in Fig. 1b.

Although providing good results without the need of a carefully chosen parameterization, this approach is computationally too expensive – being NP-hard in general – to become applied on robots with low computing power.

III. Particle History Clustering

After having investigated the already existing approaches and their drawbacks, we have developed a new pose extraction method for multimodal distributions in self-localization scenarios. This approach, which is closely linked with the sensor resetting mechanism for particle filters, is described in this section together with some possible extensions.

A. Multimodal Distributions and Sensor Resetting

Modeling a robot's pose as a set of particles offers a convenient option for considering altervative robot positions (e. g. to recover from kidnapping) through adding new samples. This can be done randomly or – much more efficiently – by considering recent sensor measurements [7] which might already provide a rough estimate of a valid robot pose, as depicted in Fig. 2. Such an insertion of particles can be done permanently by replacing a fixed number of particles per time frame or dynamically, e. g. based on the development of the overall weight of the particle set [4, 11].

In the following, we consider every insertion of a new particle as the establishment of a new hypothesis and as a starting point for the robot pose extraction from a multimodal distribution.

B. The Basic Algorithm

The approaches described in Sect. II have one thing in common: They are completely separated from the execution of the Monte Carlo algorithm, and they operate on the final sample set only. In contrast, the proposed method is partially linked into the particle filter – without modifying its functionality in any way but through adding new operations – and thus provides an elegant way to identify sample clusters representing a robot pose hypothesis.

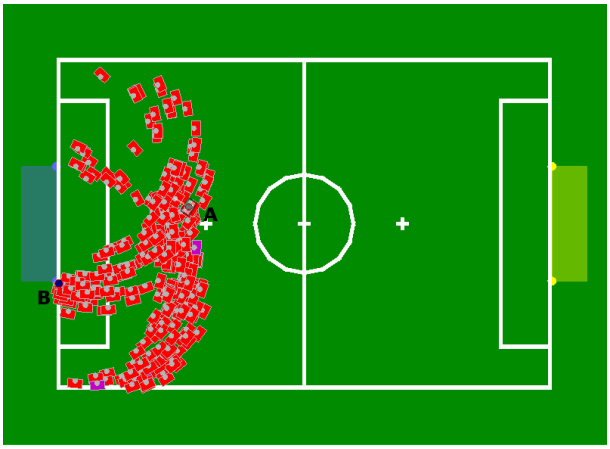

Fig. 2. Sensor resetting: A robot standing at position A observes the non-unique goalpost at position B. The red boxes denote a sample set which has been fully generated using this single observation. All samples are situated on circles around the two possible goal posts. The different distances to the posts are a result of the sensor model's uncertainty.

Initialization at startup:
for each $s \in S_t$
ASSIGNFREECLUSTERINDEX(S,C)

Inside every Monte-Carlo resampling step:
for each $s \in S_{t+1}$
if $s \in S_t$
ASSIGNCLUSTERINDEXOFPARENT(S)

After every Monte-Carlo resampling step:
for each $s \in S_t$
if $s \notin S_{t+1}$
RETURNCLUSTERINDEXTOSET(S,C)

Insertion of a new sample:
for each $s \in S_{t+1}$
if $s \notin S_t$
ASSIGNFREECLUSTERINDEX(S,C)

Fig. 3. Operations of particle history clustering.

We assume the following:

1) Generating a new sample is considered as setting up a new hypothesis (as described in Sect. III-A) and starting a new cluster respectively.
2) If a sample is copied by the Monte-Carlo resampling step, all copies of the sample belong to the same cluster as the original sample.
3) Samples do not change the hypothesis they belong to (unless this is forced, cf. III-C).

This leads to the following, straightforward approach: When initializing the particle filter, every sample is treated as a single cluster having a unique cluster identifier. During the resampling step of the filter, some samples are copied one or more time to the new sample set, some samples drop out. The copy operation also copies the cluster index. Thus, some clusters grow over time whilst some drop out. The total number of different clusters decreases over time. Without the insertion of new particles, this proceeding would converge towards a single cluster. Through sensor resetting, new samples are generated. Every new sample is again treated as a cluster and thus receives an according identifier. These samples are – of course – again processed as all others. By establishing this simple assignment mechanism that does not interfere with the basic Monte-Carlo algorithm the tracking of different hypotheses is possible.

The pose computation can now be realized through averaging all samples belonging to the largest cluster. If multiple hypotheses have to be extracted, e. g., to be processed by control modules that are able to deal with this kind of data, poses for all other existing clusters can also be computed.

A formal description of the approach is given in Fig. 3. Let S_t be the current sample set, S_{t+1} the sample set after the resampling step, and C the set of all unused cluster indices and thus possible clusters respectively:

C. Additional Merging Operations

When starting a new cluster based on an inserted sample, there is no mechanism that prevents it from being at the same position as (or very close to) an already existing cluster. Thus, different clusters might evolve in parallel describing almost the same hypothesis. In practice, this parallelism will vanish over time since one cluster will slightly fit better and thus dominate the other one step by step. This process can be accelerated by different merging operations. Nevertheless, applying merging operations is a trade-off between a fast execution of the pose extraction and a fast stabilization of the clusters.

1) Merge Random with Largest Cluster: A simple merging operation is to select a random cluster and to try to integrate it into the currently largest cluster, as specified in Fig. 4.

MERGERANDOMTOLARGEST (*Clusters*)
$C_L \leftarrow$ GETLARGEST(*Clusters*)
$C_R \leftarrow$ GETRANDOM(*Clusters*)
if not $C_L = C_R$
if COMPATIBLE(C_L,C_R)
MERGE (C_L,C_R)

Fig. 4. Random cluster merging.

This operation can, e. g., be performed once per execution of the self-localization component. Over time, all clusters will be tested for merging. If the probability distribution is completely covered by a single cluster only, this merging operation does not occur. Whilst the merging action is trivial since it is only an assignment of a new index to some samples, the compatibility test might be more complex depending on the function chosen for comparison, e. g. Mahalanobis distance, geometric enclosure, or simply a threshold for the distance of the cluster centers.

2) Merge Largest Clusters: Whilst the previous operation tests all other clusters over time and thus might lead to a

continuous integration of many small clusters into the currently largest one, merging the two largest clusters is almost the same operation but has a slightly different aim. It ignores all small clusters – which often consist of one particle only – since they do not contribute to the final result anyway. The two largest clusters are potential candidates for oscillations between slightly different results. Thus merging them – if possible – provides a more stable pose estimate. The possible compatibility tests are the same as for the previous merging scheme.

3) General Merging of Clusters: Of course, also general merging tests could be performed, i. e. every cluster is tested for being compatible to any other cluster. However, such tests would be computationally expensive and contradictory to the algorithm's original simplicity.

4) Single Sample Transfer: Going one level deeper, from clusters back to single samples, a modification of the *MergeRandomToLargest* scheme is to check single samples for their compatibility with the largest cluster and – if applicable – to change their cluster identifier accordingly. This procedure is specified in Fig. 5.

TRANSFERSAMPLETOLARGEST (*Clusters, SampleSet*)
  $C_L \leftarrow$ GETLARGEST(*Clusters*)
  $S_R \leftarrow$ DRAWRANDOMLY(*SampleSet*)
  if not $S_R \in C_L$
    if COMPATIBLE(S_R,C_L)
      ADD (S_R,C_L)

Fig. 5. Single sample transfer to the largest cluster.

D. Known Drawbacks

As any other existing approach for the given problem, also particle history clustering has some inherent drawbacks that might make it inappropriate for some self-localization scenarios.

One obvious problem is: the approach does not work if no new samples are added to the sample set. For this case, no reasonable scheme for assigning cluster indices to samples can be established.

In Sect. III-C, some possible approaches for merging clusters have been described. In some scenarios, the opposite operation might be necessary: the splitting of a cluster. When navigating freely in open space, e. g. as in a RoboCup scenario, this rarely happens and can be neglected. But when navigating along predefined routes, e. g. in an office space scenario, a split-up might be necessary when the samples of a cluster divide along different routes at a branching. Of course, analyzing and splitting operations, similar to the merging operations, could be implemented. But this is not a trivial task and would come along with a much larger demand of computing time and thus eliminate the advantages of the approach.

IV. EXPERIMENTAL RESULTS

To evaluate the proposed approach and to compare it with other common approaches for pose extraction, two different experiments – one with several robot kidnappings and one without any – have been conducted on a humanoid robot platform. The computed robot poses are compared against a globally tracked reference position.

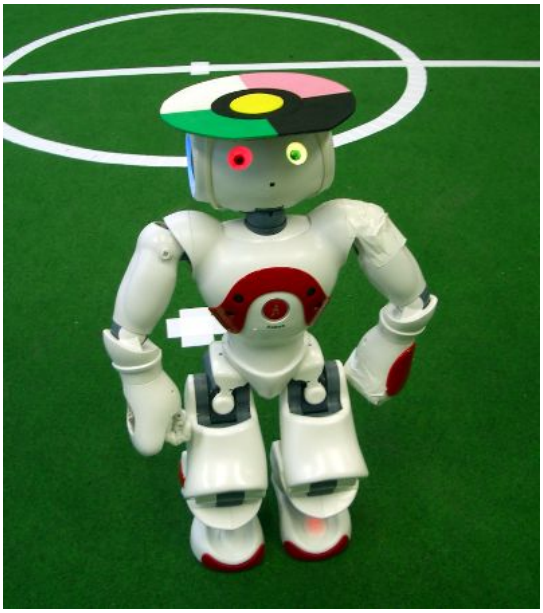

Fig. 6. A Nao RoboCup robot used for the experiments. A colored pattern for global tracking has been attached to its head.

A. Robot Platform and Environment

For the experiments, we have used the RoboCup edition of the Nao robot [3] made by Aldebaran Robotics as shown in Fig. 6. The robot is equipped with 21 degrees of freedom, a 500 MHz processor, and a camera as main sensor[1]. In addition, it provides measurements of joint angles which are combined with accelerometer and gyroscope data for body posture estimation. Image processing and self-localization are performed at the camera's frame rate, i. e. 30 Hz.

The experimental environment is a standard RoboCup Standard Platform League field as specified in [1]. The field size is $7.4m \times 5.4m$. The only unique features are two colored goals. In addition, the robot is able to perceive points on the field lines. These are non-unique features that are also be matched by the self-localization module.

The robot runs the software framework of the RoboCup team *B-Human* [10] in the same configuration as it is used during real competitions, e. g., during the team's win of the RoboCup German Open. The self-localization component is an improved version of the one described in [5] that is based on [9], additionally including the *Augmented MCL* resampling approach of [4]. The sample set of the filter has been configured to use 100 samples.

As source for ground truth data, a global tracking system has been used. For this purpose, a unique marker has been fixed on the robot's head (cf. Fig. 6) and been tracked by a

[1] In fact, the robot has two cameras but only the lower one has been used for these experiments.

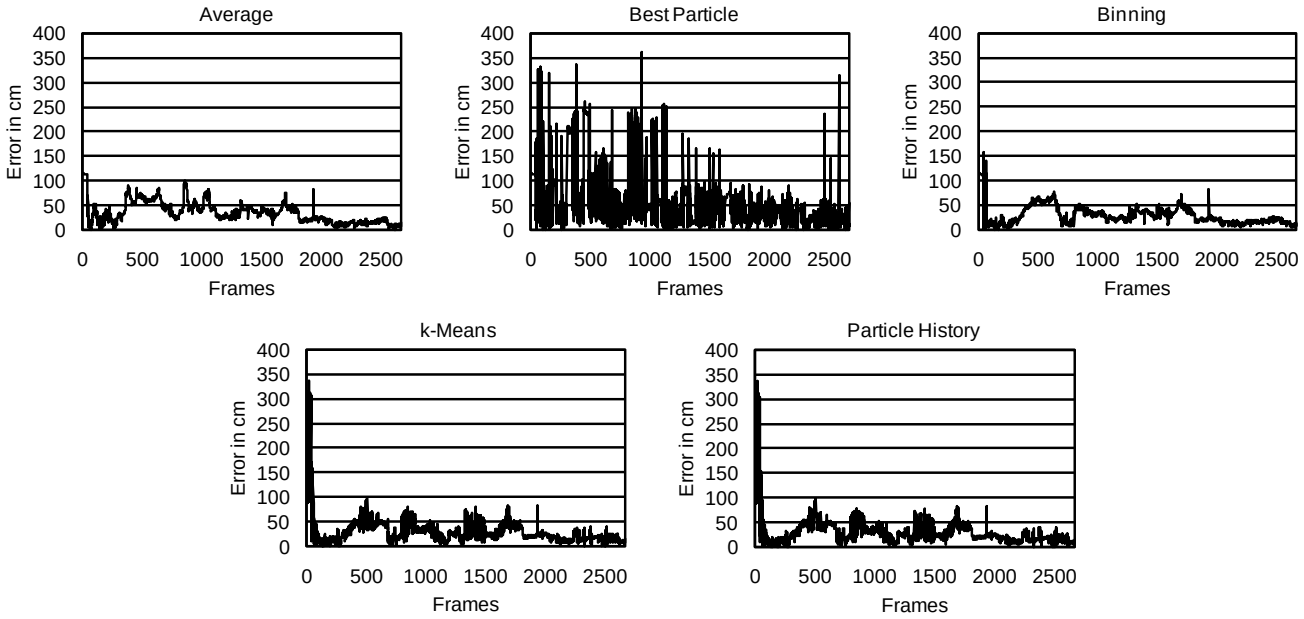

Fig. 7. Errors during position tracking. Comparision of the different approaches which were used to extract the pose of the walking robot from the particle set.

camera hanging above the field, the software for this purpose has been developed by the RoboCup Small Size team B-Smart [6]. The system provides the position as well as the rotation (which is fused with the robot's head rotation) of the robot on the field.

B. Experiment I - Continuously Walking

In this experiment, the robot has been walking omni-directionally along a predefined eight-shaped route and looking around on the field for several minutes. No robot kidnapping occurred during the experiment but nevertheless, new particles have been inserted whenever the localization quality dropped.

Figure 7 illustrates the position errors resulting from the different approaches. The average position errors are listed in Tab. I. Binning, k-means clustering, and particle history clustering seem to perform with the same precision within the experimental environment (the position after the decimal point in Tab. I can be considered as pure chance given the noise of the environment). Averaging over all particles performs slightly worse, probably due to outliers which negatively influence the results.

TABLE I

ERRORS DURING POSITION TRACKING. IGNORED THE FIRST 200 FRAMES.

Approach	Avg. in cm	Stdv. in cm
Average	34.6	19.9
Best Particle	50.1	51.6
Binning	28.7	15.3
K-Means	28.4	17.3
Particle History	28.3	17.2

C. Experiment II - Recovery from Kidnapping

In this experiment, the robot was standing and looking around. During the experiment, four kidnappings occurred in which the robot was carried to a different position and rotation. The software configuration was the same as in the previous experiment except for one parameter influencing the insertion of new samples which was configured to be more likely.

Figure 8 illustrates the position errors resulting from the different approaches. The average position errors are listed in Tab. II. Again, binning, k-means clustering, and particle history clustering seem to perform equally and the selection of the best particle performs worst. At first sight, the performance of the overall average might surprise. But as can be seen in Fig. 8, this is due to the much lower peaks within the short time frames after the kidnapping actions that result from taking all samples into account for pose computation.

TABLE II

ERRORS WITH RELOCALIZATION. IGNORED THE FIRST 200 FRAMES.

Approach	Avg. in cm	Stdv. in cm
Average	46.8	31.2
Best Particle	54.0	49.0
Binning	50.2	36.3
K-Means	50.9	39.2
Particle History	50.8	39.2

V. CONCLUSIONS

In this paper, we presented a novel approach for pose extraction from Monte-Carlo sample sets in robot self-localization scenarios. In contrast to commonly used methods, it is integrated into the standard filter algorithm and allows a very efficient tracking of multiple hypotheses in multimodal probability distributions which are likely to occur when using sensor

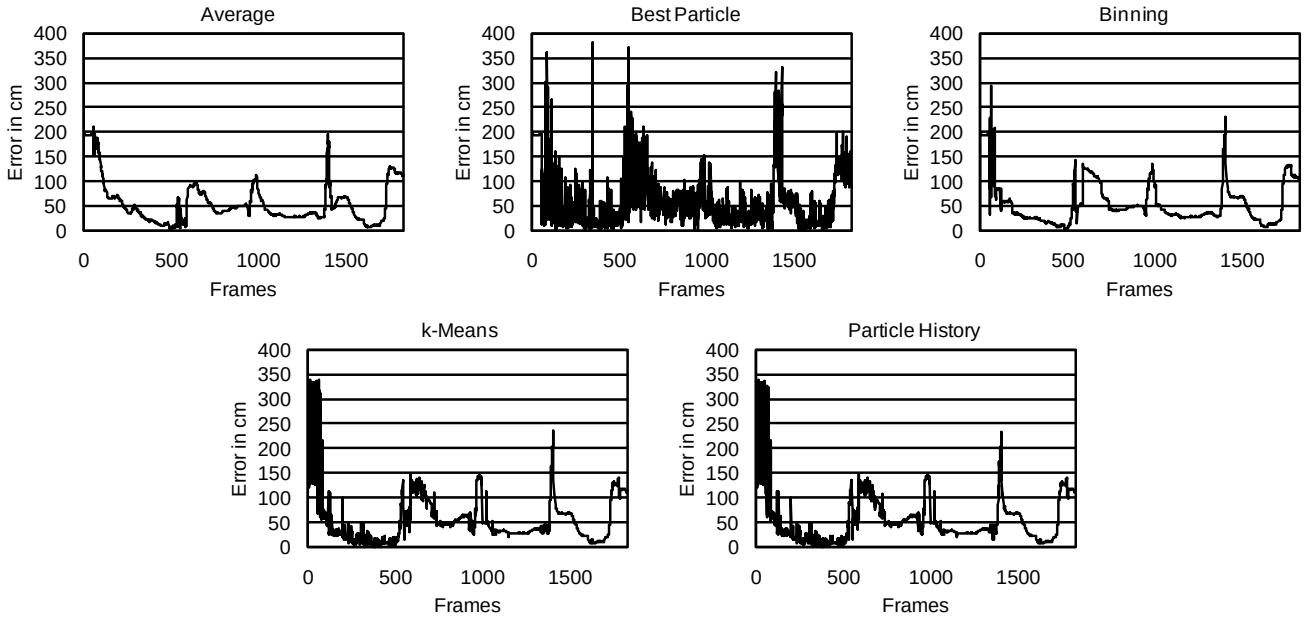

Fig. 8. Errors during relocalization. Comparision of the different approaches which were used to extract the pose of the kidnapped robot from the particle set.

resetting mechanisms to overcome problems like robot kidnapping. The algorithm does not share some of the drawbacks of others approaches but has some other specific limitations. As the real-robot experiments conducted in a RoboCup scenario show, the approach is able to provide results of a similar quality than approved – but computationally or configurationally more complex – algorithms. Thus, it appears to be an elegant and efficient alternative for some scenarios.

ACKNOWLEDGMENT

The authors would like to thank Armin Burchardt and Andreas Seekircher for their assistance during the experiments conducted as well as all B-Human team members for providing the software base for this work.

REFERENCES

[1] RoboCup Technical Committee. RoboCup Standard Platform League (Nao) Rule Book. http://www.tzi.de/spl/pub/Website/Downloads/Rules2009.pdf as of April 29th, 2009.

[2] D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte Carlo localization: Efficient position estimation for mobile robots. In *Proc. of the National Conference on Artificial Intelligence*, 1999.

[3] David Gouaillier, Vincent Hugel, Pierre Blazevic, Chris Kilner, Jerome Monceaux, Pascal Lafourcade, Brice Marnier, Julien Serre, and Bruno Maisonnier. The nao humanoid: a combination of performance and affordability. *CoRR*, abs/0807.3223, 2008.

[4] J.-S. Gutmann and D. Fox. An experimental comparison of localization methods continued. *Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2002.

[5] T. Laue and T. Röfer. Particle filter-based state estimation in a competitive and uncertain environment. In *Proceedings of the 6th International Workshop on Embedded Systems*. VAMK, University of Applied Sciences; Vaasa, Finland, 2007.

[6] Tim Laue, Armin Burchardt, Kai Cierpka, Sebastian Fritsch, Nils Göde, Kamil Huhn, Teodosiy Kirilov, Bianca Lassen, Eyvaz Lyatif, Markus Miezal, Markus Modzelewski, Ulfert Nehmiz, Malte Schwarting, Andreas Seekircher, and Ruben Stein. B-Smart - Team Description for RoboCup 2008. In L. Iocchi, H. Matsubara, A. Weitzenfeld, and Changjiu Zhou, editors, *RoboCup 2008: Robot Soccer World Cup XII Preproceedings*. RoboCup Federation, 2008.

[7] Scott Lenser and Manuela Veloso. Sensor resetting localization for poorly modeled mobile robots. In *Proceedings of the 2000 IEEE International Conference on Robotics and Automation (ICRA 2000)*, volume 2, pages 1225–1232, San Francisco, CA, USA, 2000.

[8] T. Röfer and M. Jüngel. Vision-based fast and reactive monte-carlo localization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA-2003)*, pages 856–861, 2003.

[9] T. Röfer, T. Laue, and D. Thomas. Particle-filter-based self-localization using landmarks and directed lines. In A. Bredenfeld, A. Jacoff, I. Noda, and Y. Takahashi, editors, *RoboCup 2005: Robot Soccer World Cup IX*, number 4020 in Lecture Notes in Artificial Intelligence, pages 608–615. Springer, 2006.

[10] Thomas Röfer, Tim Laue, Armin Burchardt, Erik Damrose, Katharina Gillmann, Colin Graf, Thijs Jeffry de Haas, Alexander Härtl, Andrik Rieskamp, André Schreck, and Jan-Hendrik Worch. B-Human. Team Report and Code Release 2008. http://www.b-human.de/download.php?file=coderelease08_doc as of April 29th, 2009.

[11] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005.