Detection Strategy for Kidnapped Robot Problem in Landmark-Based Map Monte Carlo Localization

I. Bukhori and Z. H. Ismail

Department of Electronic Systems Engineering,
Malaysia-Japan International Institute of Technology,
Universiti Teknologi Malaysia,
Jalan Sultan Yahya Petra, 54100, Kuala Lumpur, Malaysia.
E-mail: bukhoriiksan@gmail.com, zool@utm.my

T. Namerikawa

Department of System Design Engineering,
Keio University,
3-14-1 Hiyoshi, Kohoku-ku,
Yokohama 223-8522, Japan.
Email: namerikawa@sd.keio.ac.jp

Abstract— **This paper proposes a new method to detect the kidnapped robot problem event in Monte Carlo Localization. The method is designed such that it can provide accurate detection in wide range of particles' convergence level and does not depend too much on the re-localization/recovery process. The proposed method combines the difference in particle's weight, maximum current weight, and difference in particles' standard deviation. The addition of these two parameters is believed to be superior to a pure maximum current weight parameter for kidnapping detection. A series of simulation tests are executed to prove it. These simulations show that the proposed method outperforms the maximum current weight parameter in terms of accuracy, ability to detect kidnapping during early stage of localization, and independency towards the success of the re-localization process.**

Keywords— **Monte Carlo Localization, Kidnapping Detection, Difference of Weight, Difference of Standard Deviation, Early Kidnapping, Recovery Independency**

I. INTRODUCTION

In mobile robotics localization, the kidnapped robot problem is defined as a condition when the robot is instantly moved to other position without being told during the operation of the robot [1], [7-9]. Kidnapped robot problem is one of the most difficult problem in Monte-Carlo Localization (MCL) [2]. This is due to the nature of particle filter used in MCL itself, where the convergence process of hypotheses (particles) causes an absence of particles in some areas. This absence leads to a failure in localization if the robot is kidnapped to that area.

Kidnapped robot problem does not often happen in practice; however, it is often used to test the ability of algorithm to recover from global localization failures. Furthermore, the mechanical and sensor faults can lead to similar condition to kidnapping condition, thus the detection of this event can be used as fault detection [6].

For decades, there have been several approaches in solving kidnapped robot problem. Some solutions are based on visual recognition, such as the ones found in [4] and [5]. These approaches, however, are limited to the robot with visual-based sensor, such as camera. Some other approaches are more flexible by using the intrinsic parameters of the MCL itself. Augmented MCL proposed in [1] and MCL with mixture distributions [1], [8], and [18] are some examples of this category.

In Augmented MCL, random particles are injected in each iteration such that the possibility of particles' absence in kidnapping destination area is reduced. These random particles are drawn from either uniform distribution over pose space, or the posterior of the measurement. MCL with mixture proposal distribution combines regular MCL sampling with its dual.

Despite its flexibility, the former two methods do not clearly draw a line between detection and recovery of kidnapping. This creates a problem when the concern is not only in the re-localization, but also the needs to know when the kidnapping really happens, such as in fault detection.

Other solutions which also depend on intrinsic parameters of MCL can be found in [2] and [3]. Zhang *et.al* in [2] uses maximum weight of current particle set as the parameter to detect the kidnapping event. Yi, C. and B.-U. Choi in [3] uses similar parameter, but instead of purely using current weight, they use the *entropy* of the information can be extracted from the weight. These two approaches address the detection and recovery separately, as what we prefer.

In this paper, we use the maximum current weight (MCW) as the benchmark to compare our proposed method of kidnapping detection with. This choice is due to the fact that the approach in [3] rises from the same parameter, which is current weight. Therefore, it is arguably similar to MCW in terms of the drawbacks.

This paper introduces a new method which mixes MCW with mean weight parameter and standard deviation of particles. The combination of these three parameters arises as an attempt to increase the accuracy of the detection.

The rest of the paper is organized as follows. In chapter II Bayes filter and Monte Carlo Localization are briefly explained. Chapter III explains the definition of some terms used throughout the paper. Chapter IV reviews MCW method for kidnapping detection. The method we propose is then delivered in detail in Chapter V while chapter VI explains the simulations result of the comparison between the proposed method and MCW. Finally, chapter VII gives the conclusion.

978-1-4673-7124-7/15/$31.00 ©2015 IEEE

II. BAYES FILTER AND MONTE CARLO LOCALIZATION

In mobile robot localization practice, it is almost impossible for a robot to know exactly its coordinates and heading (collectively known as *pose*) in the given map. Rather, the robot should infer them from the data from environment. The obtained state is then called *belief*. The belief of the robot is defined as

$$bel(S_t) = p(s_t|z_{1:t}, u_{1:t}) \quad (1)$$

This posterior is the probability distribution over the state $s_t = \langle x_t, y_t, \theta_t \rangle$, at time t, given all past measurements $z_{1:t} = \{z_1, z_2, \dots, z_t\}$ and all past controls $u_{1:t} = \{u_1, u_2, \dots, u_t\}$. Sometimes it is also useful to consider the belief *before* the current measurement taken, that is

$$\overline{bel}(s_t) = p(s_t|z_{1:t-1}, u_{1:t}) \quad (2)$$

MCL is a Bayesian-based localization algorithm. It provides a powerful tool to calculate posterior $bel(*)$, given measurement and control data [1], [11]. Bayes filter is based on Markov world assumption, i.e. past and future data are independent if one knows the current state s_t [1]. By implementing Bayes rule and this Markov world assumption, the belief posterior can be defined as

$$bel(s_t) = \eta p(z_t|s_t)\, \overline{bel}(s_t) \quad (3)$$

The term $p(s_t|s_{t-1}, u_{1:t})$ is defined as the *prediction* or *motion model*, since it reflects the state transition due to robot motion. The probability $p(z_t|s_t)$ itself is called *correction* or *sensor model*, since it incorporates sensor reading to update robot state. η is normalization constant, ensuring the final result to be normalized to one.

Bayes filter gives freedom to the choices of representation for the posterior. MCL represents the posterior $bel(s_t)$ by a set S_t of N weighted samples distributed according to the posterior [1], [3]. The density of the samples proportionally represents the likelihood of the robot's pose being there.

$$S_t = \langle s_t^{[n]}, \omega_t^{[n]} \rangle; \quad n = 1,2, \dots, N \quad (4)$$

Each particle $s_t^{[n]}$ represents the hypothesis of the robot's pose at time t. The ω_t^n is the non-negative number called weight of particle. It indicates how good particle $s_t^{[n]}$ in representing the robot's pose.

The basic MCL algorithm, as summarized from [1] and [2] is depicted in table 1. It accepts previous state S_{t-1}, past controls u_t, past measurements z_t, and map information m.

Table 1 MCL Algorithm

1. **MCL Algorithm** (S_{t-1}, u_t, z_t, m)
2. $S_t = \bar{S}_t = \emptyset$
3. **for** n = 1 to N **do**
4. generate $s_t^{[n]} \sim p\left(s_t \middle| s_{t-1}^{[n]}, u_{1:t}, m\right)$
5. calculate weight $\omega_t^{[n]} = p\left(z_t \middle| s_t^{[n]}, m\right)$
6. $\bar{S}_t = \bar{S}_t + \langle s_t^{[n]}, \omega_t^{[n]} \rangle$
7. **end for**
8. normalize ω_t
9. **for** n = 1 to N **do**
10. draw $s_t^{[n]}$ with probability $\propto \omega_t^{[n]}$
11. Add $s_t^{[n]}$ to S_t
12. **end for**
13. **return** S_t

III. DEFINITION OF THE TERMS IN KIDNAPPING

Before going further, we have to determine the definition of kidnapping detection and criteria of successful detection. We define kidnapping detection as the process detecting the time t, where $1 \leqslant t \leqslant T$, of when kidnapping happens. It does not detect the place where the robot is being kidnapped or where it is kidnapped to. Kidnapping point is defined as the time instance t of when the robot is kidnapped. We also define the criteria of successful detection as follows;

1. The detection should occur only once, since we consider single kidnapping event only.
2. The time of detected kidnapping should be the same as the real kidnapping.

These two criteria are used to compare the accuracy between MCW and the proposed method. Lastly, the recovery/re-localization process is defined as a method to localize the robot after kidnapping event. We will use recovery and re-localization interchangeably in this paper.

IV. MAXIMUM CURRENT WEIGHT METHOD

The detection in MCW can be denoted by the following equation;

$$Kidnapped_t = \omega_t^{max} < \xi \quad (5)$$

Where ω_t^{max} is the maximum weight of the particle set at time t, while ξ is the threshold to detect the kidnapping.

This formulation leads to two problems;

A. Early Kidnapping Detection

If the robot is kidnapped early, i.e. when the particles are still spread, the weight of particles will still be quite high. This is due to the fact that it is very likely there would be some particles at/near where the robot is being taken to (see Fig. 1). Therefore, the kidnapping around this time step will be very

likely undetected. In Figure 1, the robot is being kidnapped at t=3. There is no recovery strategy applied, but we can see the robot can still re-localize itself as shown by the constant maximum weight of 1. This is again due to the existence of particles near the point where the robot is after being kidnapped.

As can be seen in Fig 1, the maximum weight at t = 3 is still high. This causes the inability of Eq. 5 to detect the kidnapping. Forcing this equation to detect the kidnapping at t = 3 by setting $\xi > \omega_3^{max}$ will make multiple detection outside t = 3, which means the detection fails as what discussed in chapter III. Kidnapping in this part (early kidnapping) does not affect the overall localization too much. However, sometimes an information of when the kidnapping happens can be used as fault detection such that one can know something unnatural happens around the kidnapping point (t=3 in the example). MCW obviously fails at obtaining this information.

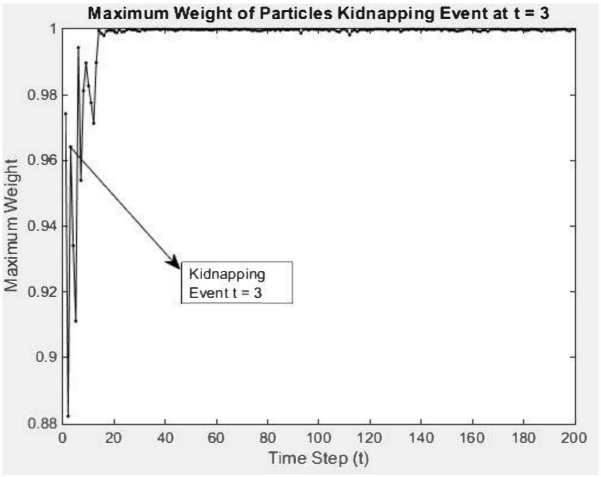

Fig 1 Maximum Weight of Particles with Kidnapping Event Happens at t = 3.

B. *High Dependency to The Success of Recovery*

There is also a high dependency between current weight and the ability of the recovery strategy. The weight of particle is based on how close the observation from particle to the observation from the robot. Therefore, when the recovery fails, the gap between two observations will be high and thus resulting in lower particles' weight (see Fig. 2).

From the figure we can see that when there is no recovery strategy (extreme case of recovery failure) to re-localize the robot, the maximum weight drops very low. Forcing the detection to detect kidnapping will result in multiple detections at later time steps, thus the failure in detection.

The two drawbacks in MCW give motivation to devise a new detection strategy which is able to detect the kidnapping with the following nature:

a) The detection should be able to detect kidnapping in large range of particle convergence level
b) The detection should be independent to the success of the recovery, i.e. detection of kidnapping with high accuracy can still be obtained when the recovery fails or no recovery is implemented.

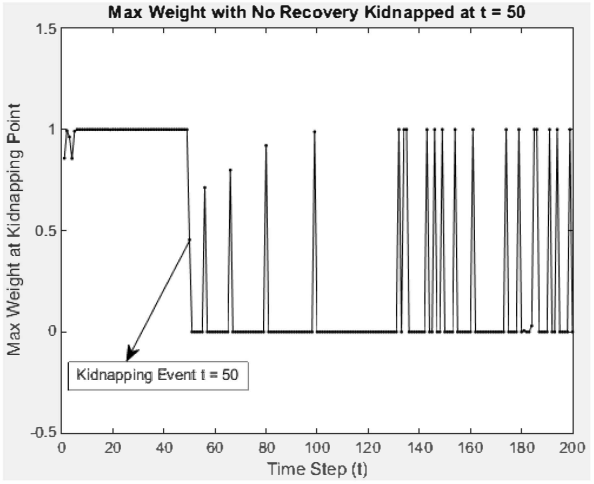

Fig 2 Max Weight of Particles with Kidnapping Event at t =50 and no Recovery

V. PROPOSED METHOD

In order to overcome the drawbacks in Max Current Weight (MCW) approach, a novel detection algorithm is proposed. Rather than using only one parameter, we add two more parameters to detect the kidnapping event; change in weight and change in standard deviation.

The two parameters are defined in eq.6 and eq.7

$$\Delta\bar{\omega}_t = \begin{cases} \bar{\omega}_t, & t = 1 \\ \bar{\omega}_t - \bar{\omega}_{t-1}, & t > 1 \end{cases} \quad (6)$$

$$\Delta\bar{\sigma}_t = \begin{cases} \bar{\sigma}_t, & t = 1 \\ \bar{\sigma}_t - \bar{\sigma}_{t-1}, & t > 1 \end{cases} \quad (7)$$

Where $\bar{\omega}_t = mean\{\omega_t^1, \omega_t^2, \dots, \omega_t^N\}$ is the mean of particles at time instance t and $\bar{\sigma}_t = mean\{\sigma_t^x, \sigma_t^y, \sigma_t^\theta\}$ is the mean of standard deviation of all three pose components. In order to further improve the accuracy, we also implement a simple convergence state detector. This detector is expressed as

$$D_t = \begin{cases} 1, & N_c > \varphi \\ 0, & elsewhere \end{cases} \quad (8)$$

Where N_c denotes the percentage of the number of particles exist within a unit distance from the mean. The constant φ is the threshold to decide whether the particles is in convergence state or not. In our experiment we use $\varphi = 70$. Based on this number, this detector divides the detection into two areas; pre-convergence/early kidnapping and post-convergence/late kidnapping.

A. *Early Kidnapping*

We define early kidnapping (EK) as a kidnapping event which happens inside pre-convergence area ($D_t = 0$). In this area, kidnapping will not cause significant change in weight parameter due to the particles existence around the map. However, the high weighted particles near kidnapping destination will 'pull' other particles towards them, causing a small rise in $\Delta\bar{\sigma}_t$.

During recovery, however, the similar situation may occur. In order to prevent misdetection cause by recovery, we apply two simple rules which are:

a. Pre-convergence state always happens before post-convergence state.
b. After EK is detected, no further detection is valid before the particles reach convergence state.

These rules are implemented by assigning $D_t = 1$ when kidnapping is detected at time *t* and define the early kidnapping as:

$$Kid_{Ek(t)} = \left(\neg \sum_{1}^{t} D_t\right) \wedge (\Delta\bar{\sigma}_t > \beta) \tag{9}$$

Where β is a positive small constant ($0 < \beta < 0.1$) which defines the sensitivity of EK detection.

B. *Late Kidnapping*

As opposed to early kidnapping, we define late kidnapping (LK) as the kidnapping which happens when $D_t = 1$. In this post-convergence area, kidnapping event causes a significant drop in $\Delta\bar{\omega}_t$. We define the kidnapping in this area as:

$$Kid_{Lk(t)} = D_t \wedge (\omega_t^{max} < \xi) \wedge (\Delta\bar{\omega}_t < \alpha) \tag{10}$$

In step 8 of MCL algorithm (Table 1), the weight of particle is normalized. Therefore the maximum possible weight at each iteration is $\omega_t^{max} = 1$. Based on the maximum possible weight and the fact that $\Delta\bar{\omega}_t < 0$ at kidnapping event, we can conclude the range for α is $-1 < \alpha < 0$ while $0 < \xi < 1$.

The first term in eq.10 is the convergence criteria. The second term detects the kidnapping as what the MCW method does while the last term is the insurance against failure of recovery because in the case of failed recovery, the first and second term will still be satisfied but the change in weight will not be significant thus avoiding misdetection.

C. *Complete Algorithm for the Proposed Detection Method*

After defining the condition of kidnapping in two scenarios, early and late kidnapping, we have to inject the algorithm to the MCL algorithm shown in Table 1 with some adjustment to improve the method.

A modified MCL algorithm with kidnapping detection and recovery routine is shown in Table 2.

Table 2 Modified Monte Carlo Localization

```
1.  MCL Algorithm (S_{t-1}, u_t, z_t, m)
2.  S_t = S̄_t = D = ∅
3.  for n = 1 to N do
4.      generate s_t^[n] ~ p(s_t | s_{t-1}^[n], u_{1:t}, m)
5.      calculate weight ω_t^[n] = p(z_t | s_t^[n], m)
6.      S_t = S_t + ⟨s_t^[n], ω_t^[n]⟩
7.  end for
8.  normalize ω_t
9.  for n = 1 to N do
10.     draw s_t^[n] with probability ∝ ω_t^[n]
11.     Add s_t^[n] to S_t
12. end for
13. define convergence state D_t
14. add D_t to D
15. calculate σ̄_t and Δω̄_t
16. K_t = Kidnap Detection (t, D, Δσ̄_t, Δω̄_t)
17. if K_{t-1} = True
18.     Recovery Strategy
19. end if
20. K_{t-1} = K_t
21. return S_t
```

In Table 2, lines 15-18 show that the recovery process takes place at the time step right after the detected kidnapping point, instead of at the same time instance as kidnapping point. This is because in recovery process, the parameters we described before might trigger the kidnapping detection, such as the increase of standard deviation. Therefore, the separation between detection and recovery initialization is needed.

The details of the proposed detection algorithm called by line 14 is shown in Table 3.

Table 3 Algorithm of Proposed Method for Kidnapping Detection

```
1.  Kidnap Detection (t, D, σ̄_t, Δω̄_t)
2.  if t = 1
3.      α = α_init; β = β_init; ξ = ξ_init
4.  end if
5.
6.  EKflag = (¬ Σ_1^t D_t) ∧ (Δσ̄_t > β)
7.  LKflag = D_t ∧ (ω_t^max < ξ) ∧ (Δω̄_t > α)
8.
9.  if EKflag ∨ LKflag
10.   if Ekflag
11.       D_t = 1
12.   end if
13.   return True
14. else
15.   return False
16. end if
```

VI. SIMULATION RESULT

A series of simulation is tested against the MCW and the proposed method to see which of the two is closer to the desired nature of kidnapping detection stated at the end of chapter IV. Each test is run using 15x15 landmark-based map with 10 randomly-placed landmarks. There are 200 time instance of kidnapping, $t_k = \{1,2,3,\ldots,200\}$. For each t_k, the simulation with 200 time steps is run for 100 times.

The number of successful kidnapping detection is then calculated for each t_k, and divided by 100 to obtain the percentage of success rate. An example of the map used and the desired trajectory of the robot is depicted in Fig. 3. All simulations will use the same robot path.

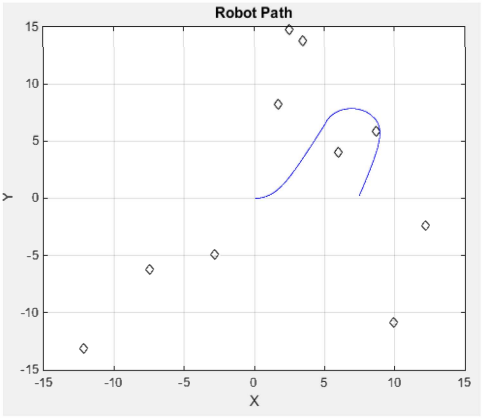

Fig. 3 One Instance of the Map and The Desired Trajectory of The Robot

The kidnapping event is expressed as follows:

$$s_{t_k} = s_{t_{k-1}} + [9 \quad 9 \quad -0.5\pi] \tag{12}$$

1) Detection Test Without Recovery

In order to test the independency of the two methods towards recovery, a kidnapped robot problem is simulated without recovery process. The test result is depicted in Fig. 4

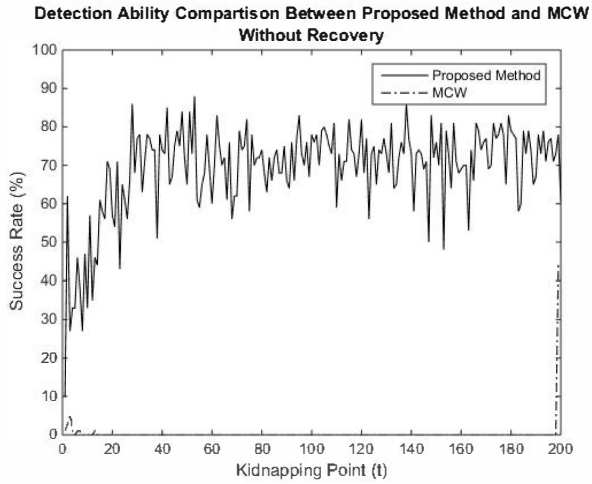

Fig. 4 Success Rate per Kidnapping Point without recovery

In Fig. 4 we can see that the proposed method clearly outperforms MCW in all of the kidnapping points. It shows that even without recovery, the kidnapping can still be detected with high accuracy. On the other hand, pure MCW method could not detect the kidnapping except when it happens at $t=200$. It proves our discussion in chapter IV about recovery dependency.

2) Detection Test With Recovery

Because the MCW fails in detection without recovery, other type of test is devised. In this test, the recovery strategy is implemented at the time step right after the kidnapping point $(t_{recovery} = t_k + 1)$ regardless of the success of the detection of either method. The recovery strategy employed is the most basic one, which is the particles' re-initialization method. This recovery process is executed by replacing current set of particles by the randomly distributed particles drawn from uniform distribution over the pose space inside the map, the same as particles initialization at the very beginning of localization process. The same value of ξ is used for both method. How the two methods behave under this test is depicted in Fig. 5

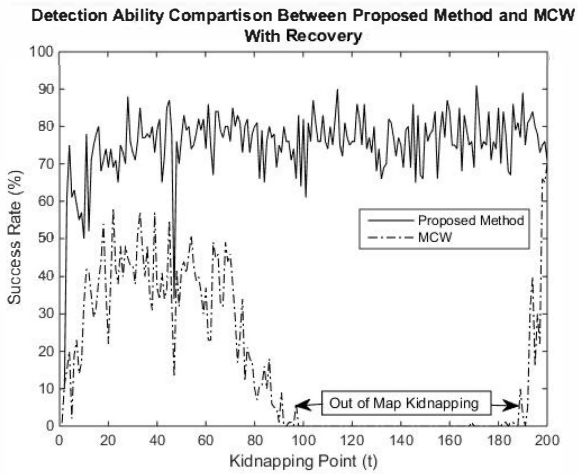

Figure 5 Success Rate per Kidnapping Point with Recovery

The result in Fig. 5 shows that proposed method still outperforms the MCW. This includes the kidnapping event in pre-convergence state (t<10). In $100 < t < 190$ it can be seen that MCW even fails to correctly detect the kidnapping. This is due to the fact that the robot is kidnapped outside of the map (see Fig. 3 and eq. 12) and the recovery strategy is only applied inside the map.

VII. CONCLUSIONS

A new method in detecting the kidnapping event in Monte Carlo Localization is proposed. The method relies on three parameters, the change in weight of particles, the change in standard deviation of the particles, and the maximum current weight. A series of simulation tests comparison between the proposed method and pure MaxCurrent Weight (MCW) is also conducted. The results show that the proposed method can still

detect the early kidnapping event with high accuracy when the MCW fails. This ability of kidnapping detection even when the recovery process itself is unneeded (characteristic of early kidnapping) shows that the proposed method is able to be a good fault detector independent to the needs of recovery.

The proposed method is also able to maintain high accuracy under the failure in recovery, showing that the independency towards the success of the recovery is very high compared to MCW. This independency is important to make sure that the information of when kidnapping happens is not disturbed by the recovery process.

ACKNOWLEDGEMENTS

This work was supported in part by the Japan-ASEAN Integration Fund (JAIF) and AUN/SEED-Net under Grant no. R.J130000.7309.4B156 and Universiti Teknologi Malaysia under Grant no. Q.J130000.2509.05H54

REFERENCES

[1] S. Thrun, W. Burgard and D. Fox, *Probabilistic Robotics* (MIT Press, Cambridge, MA, 2005).

[2] L. Zhang, R. Zapata and P. Lépinay, *"Self-adaptive Monte Carlo localization for mobile robots using range finders,"* Robotica 30, pp.229–244 (2012).

[3] Yi, C. and B.-U. Choi, *Detection and Recovery for Kidnapped-Robot Problem Using Measurement Entropy, in Grid and Distributed Computing*, T.-h. Kim, et al., Editors. 2011, Springer Berlin Heidelberg. p. 293-299.

[4] Majdik, Andras, et al. "New approach in solving the kidnapped robot problem." *Robotics (ISR), 2010 41st International Symposium on and 2010 6th German Conference on Robotics (ROBOTIK)*. VDE, 2010.

[5] Andreasson, Henrik, André Treptow, and Tom Duckett. "Localization for mobile robots using panoramic vision, local features and particle filter." *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*. IEEE, 2005.

[6] Duan, Z.; Cai, Z.; Min, H. Robust Dead Reckoning System for Mobile Robots Based on Particle Filter and Raw Range Scan. Sensors 2014, 14, 16532–16562.

[7] T. Hester and P. Stone, "Negative Information and Line Observations for Monte-Carlo Localization," *Proceedings of IEEE International Conference on Robotics and Automation ICRA 2008,* Pasadena, CA (2008) pp. 2764-2769.

[8] S. Thrun, D. Fox, W. Burgard and F. Dellaert, "Robust Monte Carlo Localization for Mobile Robots," *Artif. Intell.* **128** (1-2), 99-141 (2001).

[9] H. Andreasson, A. Treptow and T. Duckett, "Localization for Mobile Robots Using Panoramic Vision, Local Features and Particle Filter," *Proceeding of IEEE International Conference on Robotics and Automation (ICRA 05),* Barcelona, Spain (2005) pp. 3348-3353.

[10] A. R. Vahdat, N. NourAshrafoddin and S. S. Ghidary, "Mobile Robot Global Localization Using Differential Evolution and Particle Swarm Optimization," *IEEE Congress on Evolutionary Computation (CEC07),* Singapore (2007) pp. 1527-1534.

[11] S. I. Roumeliotis and G. A. Bekey, "Bayesian Estimation and Kalman Filtering: A Unified Framework for Mobile Robot Localization," *Proceedings of IEEE International Conference on Robotics and Automation (ICRA '00),* San Francisco, CA (2000) vol.3, pp. 2985-2992.

[12] S. Thrun, M. Beetz, M. Bennewitz, W. Buggard, A. Cremers, F. Dellaert, D. Fox, D. Hähnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz, "Probabilistic Algorithms and The Interactive Museum Tour-Guide Robot Minerva," *Int. J. Robot. Res.* **19**, 972-999 (2000).

[13] L. Marchetti, G. Grisetti, and L. Iocchi, "A Comparative Analysis of Particle Filter Based Localization Methods," *RoboCup*, Bremen, Germany (2006), pp. 442-449.

[14] D. Fox, W. Burgard, H. Kruppa, and S. Thrun, "Efficient Multi-Robot Localization Based on Monte Carlo Approximation," In *Robotic Research: The Ninth International Symposium* (J. Hollerbach and D. Koditschek, Eds.) (Springer-Verlag, London, 2000).

[15] C. Kwok, D. Fox and M. Meila, "Real-Time Particle Filters," *Proceedings of the IEEE,* vol. 92, no. 3, pp. 469-484, 2004.

[16] P. Pfaff, W. Burgard and D. Fox, "Robust Monte-Carlo Localization Using Adaptive Likelihood Models," *European Robotics Symposium,* Palermo, Italy (Springer Verlag, 2006) pp. 181-194.

[17] C. Siagian and L. Itti, "Biologically-Inspired Robotics Vision Monte-Carlo Localization in the Outdoor Environment," *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems IROS2007,* San Diego, CA, (2007) pp. 1723-1730.

[18] S. Thrun, D. Fox, and W. Burgard, "Monte Carlo localization with mixture proposal distribution," in Proceedings of the AAAI National Conference on Artificial Intelligence, 2000, pp. 859–865.