Design of Localization System Based on Particle Filter Algorithm for Mobile Soccer Robot Using Encoders, Compass, and Omnidirectional Vision Sensor

Ahmad Wahrudin, Augie Widyotriatmo, and Endra Joelianto

***Abstract*— Self-localization is the basis to navigate robot or vehicle in dynamic environment such as for motion planning and obstacles avoidance. Self-localization can be divided into two categories: Local Localization-System (LLS) and Global Localization-System (GLS). Local Localization-System uses inertial sensors such as encoders which leads inevitably to the unbounded accumulation of errors. Whereas Global Localization-System utilizes information based on absolute sensors so that it has a long sample time. In the Middle Size Soccer Robots, the sensors must be mounted on the robot so that it is difficult to obtain a global position directly. The particle filter algorithm is designed as a technique for combining both inertial and absolute sensor data to overcome the problems of Local Localization-System and Global Localization-System on mobile soccer robot. In this paper, three encoders are used to provide odometry motion model, an omnidirectional vision sensor is used to give weight to the particles, and ambiguity problems is overcome by using an electronic compass. The result of this test show that localization by using Particle Filter Algorithm gives better performance than Local Localization-System and can overcome the Global Localization-Problems.**

***Index Terms*—GLS, LLS, Particle Filter, Self-Localization.**

I. Introduction

MIDDLE Size League (MSL) is one of the branches in RoboCup which competes teams of five fully autonomous wheeled robots to play soccer using FIFA's sized soccer ball. All the robots must truly represent human players, so they must be able to perceive the environment through sensors that must be installed on-board. The research is focused on autonomous multi-robot control, mechatronic, multi-agent cooperation, robot perception and navigation [1][2].

Manuscript received September 15, 2019.

This work was supported in part by Engineering Physic Department, Instrumentation and Control Research Group, Institut Teknologi Bandung, the Autonomous Vehicle Research Group, and the Dagozilla Mobile Soccer Robotics Team.

A. Wahrudin was with the Department Engineering Physics, Faculty of Industrial Technology, Institut Teknologi Bandung, Bandung (e-mail: ahmadwahrudin@gmail.com).

A. Widyotriatmo is with the Instrumentation and Control Research Group, Institut Teknologi Bandung, Bandung, (e-mail: augie@tf.itb.ac.id).

E. Joelianto is with the Instrumentation and Control Research Group, Institut Teknologi Bandung, Bandung, Indonesia (e-mail: ejoel@tf.itb.ac.id).

Self-localization is one of the most important issues in autonomous mobile robots, especially for the robots in the MSL competition that has high dynamic environment [3]. Self-localization can be divided into two categories: Local Localization-System and Global Localization-System. Local Localization-System uses inertial sensors such as encoders which leads inevitably to the unbounded accumulation of errors [4] [5]. Whereas Global Localization-System utilizes information based on absolute sensors such as Global Positioning System (GPS) and active beacons so that it has more accurate measurement results compared to the Local Localization-System method but requires a longer computational time [2].

This paper presents the design and implementation of particle filter algorithm include how to characterize the noise source, that is critical to obtain better performance of MSL Soccer Robot self-localization system. This article shows encoder modification to reduce slippage error and improve the sensor reading accuracy. Information generated from encoders is used as input to the motion model while compass and omnidirectional vision sensor is used as input to the measurement model in the particle filter algorithm.

II. Modified Odometry Sensor

Odometry is the most commonly used as Local Localization-system. Odometry is obtained by calculating the incremental rotation of the wheel connected to the encoder according to its kinematics configuration. Because it only uses wheel rotation, there will always be an increase in reading errors continuously. This error is divided into 2 categories, systemic and non-systemic errors. *Systematic errors* are errors caused by imperfections in robot mechanics, such as differences in wheel diameter and unequal wheel distance to the center of the robot's geometry. *Non-systematic errors* are caused by wheel-floor interactions such as slips, bumps, and cracks [6].

In Robocup MSL, the main cause of non-systematic errors is the slip that occurs when changes in speed and direction of the robot motion [7]. A modification has been made to reduce this error by separating the encoder from the main wheel, then making it flexible by using a spring to ensure that the encoder wheel will always contact the ground as shown in Figure 1. This modification significantly reduce the accumulated errors of the system.

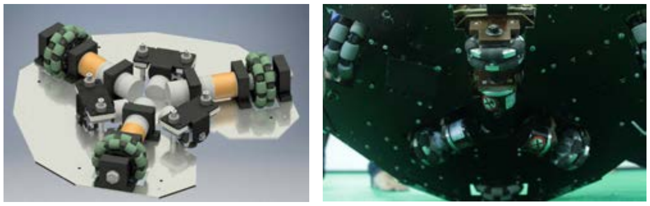


Fig. 1. Design (left) and Implementation (right) of Modified Odometry Sensor

III. Robot Kinematics

The basic modelling for our robot is shown in Fig. 2 which shows our robot model for three encoder wheels configuration with following notation [8] [6]:

- x, y, θ : relative position of the robot in meter (x, y) and angle in radian that defines the robot's heading (θ) according to the field coordinate;
- L : Distance between wheels and center of robot's geometry in meter;
- v_1, v_2, v_3 : Encoder wheels linear velocity in m/s;
- $\omega_1, \omega_2, \omega_3$: Encoder wheels angular velocity in rad/s;
- v_x, v_y : Robot linear velocity in m/s;
- ω : Robot angular velocity in rad/s

Therefore, the linear velocity vector of the encoder wheel can be represented as a matrix function of the robot's linear and angular velocity as shown (1)

$$\begin{bmatrix} v_1(t) \\ v_2(t) \\ v_3(t) \end{bmatrix} = \frac{1}{r} \begin{bmatrix} \frac{\sqrt{3}}{2} & -\frac{1}{2} & L \\ 0 & 1 & L \\ -\frac{\sqrt{3}}{2} & -\frac{1}{2} & L \end{bmatrix} \begin{bmatrix} v(t) \\ v_n(t) \\ \omega(t) \end{bmatrix} \quad (1)$$

Linear and angular velocities of the robot motion can be written as shown in (2) by calculating inverse matrix in (1).

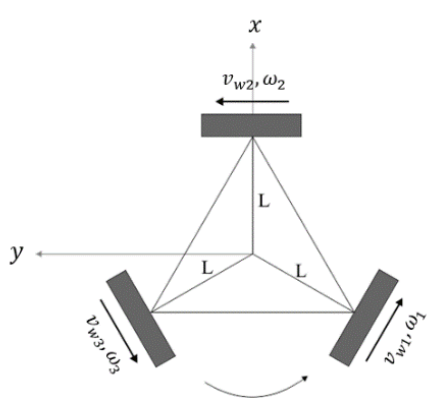


Fig. 2. Three-wheeled robot configuration model

$$\begin{bmatrix} v_x(t) \\ v_y(t) \\ \omega(t) \end{bmatrix} = \frac{1}{r} \begin{bmatrix} \frac{1}{\sqrt{3}} & 0 & -\frac{1}{\sqrt{3}} \\ -\frac{1}{\sqrt{3}} & \frac{2}{3} & -\frac{1}{3} \\ \frac{1}{3L} & \frac{1}{3L} & \frac{1}{3L} \end{bmatrix} \begin{bmatrix} v_1(t) \\ v_2(t) \\ v_3(t) \end{bmatrix} \quad (2)$$

By integrating (2), we can calculate current position and heading of the robot as shown in (3).

$$\begin{aligned} x &= \int \frac{1}{r} \left(\frac{1}{\sqrt{3}} v_1(t) - \frac{1}{\sqrt{3}} v_3(t) \right) dt \\ y &= \int \frac{1}{r} \left(\frac{1}{\sqrt{3}} v_1(t) + \frac{2}{3} v_2(t) - \frac{1}{3} v_3(t) \right) dt \\ \theta &= \int \frac{1}{r} \left(\frac{1}{3L} v_1(t) + \frac{1}{3L} v_2(t) + \frac{1}{3L} v_3(t) \right) dt \end{aligned} \quad (3)$$

By transforming the relative positions from (2), we can estimate the global position of the robot according to equation 4.

$$\begin{bmatrix} X_{new} \\ Y_{new} \end{bmatrix} = \begin{bmatrix} X_{prev} \\ Y_{prev} \end{bmatrix} + \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad (4)$$

IV. Particle Filter

Particle filter, also known as a Sequential Monte Carlo (SMC) method is an implementation of Bayes Filter which uses a Monte Carlo approach to represent the probability of a stochastic system at the present time in a Markov process. Each particle represents one of the hypotheses of the system state parameter. Each particle undergoes evolution and weighting based on a motion model and its measurement model with a certain error distribution [9].

The input of the particle filter algorithm is the set of particle $\chi_{t-1} = \{ \chi_{t-1}^{[1]}, \chi_{t-1}^{[2]}, ..., \chi_{t-1}^{[M]} \}$, the latest input control u_t , and the latest measurement z_t [9]. The particle filter algorithm is shown in Fig. 3.

```
Algorithm Particle_filter(χ_{t-1}, u_t, z_t):
    χ̄_t = χ_t = ∅
    for m = 1 to M do
        sample x_t^[m] ~ p(x_t | u_t, x_{t-1}^[m])
        w_t^[m] = p(z_t | x_t^[m])
        χ̄_t = χ̄_t + ⟨x_t^[m], w_t^[m]⟩
    endfor
    for m = 1 to M do
        draw i with probability ∝ w_t^[i
        add x_t^[i] to χ_t
    endfor
    return χ_t
```

Fig. 3. Particle Filter Algorithm

The particle filter algorithm cosists of several main process:

- Predicting process: predict new distributions for particles according to the robot motion model $P(x_t \mid u_t, x_{t-1})$;

- Updating process: update the particle weights $w_t^{[m]}$ using information from sensor model $P\left(z_t \mid x_t\right)$ and then normalize the results: the localization result can be obtained by calculating the weighted mean over all particles.
- Resampling process: acquire a new set of χ_t according to particle weights: the probability for each particles $x_t^{[m]}$ to be resampled proportional to its weight $w_t^{[m]}$.

A. *Odometry Motion Model*

The odometry motion model algorithm is used as a source of motion model in prediction step. The input of this algorithm is previous state value χ_{t-1}, and the input of the movement $u_t = \left(\delta_{trans}, \delta_{rot1}, \delta_{rot2}\right)$ [9]. Where δ_{trans} is the lateral translation of the robot movement while δ_{rot1} and δ_{rot2} are the rotational movements taken by the robot shown by Fig. 4.

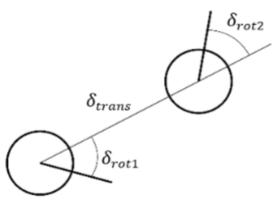


Fig. 4. Odometry Model Ilustration in Robot Linear Motion

In the sample odometry motion model algorithm for omnidirectional wheel that has been modified from common odometry motion model [9] as shown in Fig. 5 there exist 5 parameters that represent noise parameter in the robot motion model. These parameter are listed below:

- α_1: Specifies the expected noise in odometry's rotation estimate from the rotational component of the robot's motion;
- α_2: Specifies the expected noise in odometry's rotation estimate from translational component of the robot's motion;
- α_3: Specifies the expected noise in odometry's translation estimate from the translational component of the robot's motion;
- α_4 : Specifies the expected noise in odometry's translation estimate from the rotational component of the robot's motion;
- α_5 : Translation-related noise parameter that caused by omnidirectional robot characteristic.

Algorithm sample_motion_model_odometry(u_t, x_{t-1}):

$$\delta_{rot1} = atan2(\bar{y}' - \bar{y}, \bar{x}' - \bar{x}) - \bar{\theta}$$
$$\delta_{trans} = \sqrt{(\bar{x} - \bar{x}')^2 + (\bar{y} - \bar{y}')^2}$$
$$\delta_{rot2} = \bar{\theta}' - \bar{\theta} - \delta_{rot1}$$

$$\hat{\delta}_{rot1} = \delta_{rot1} - \textbf{\textit{sample}}(\alpha_1|\delta_{rot1}| + \alpha_2\delta_{trans})$$
$$\hat{\delta}_{trans} = \delta_{trans} - \textbf{\textit{sample}}(\alpha_3\delta_{trans} + \alpha_4(|\delta_{rot1}| + |\delta_{rot2}|) + \alpha_5\delta_{trans})$$
$$\hat{\delta}_{rot2} = \delta_{rot2} - \textbf{\textit{sample}}(\alpha_1|\delta_{rot2}| + \alpha_2\delta_{trans})$$

$$x' = x + \hat{\delta}_{trans}\cos(\theta + \hat{\delta}_{rot1})$$
$$y' = y + \hat{\delta}_{trans}(\theta + \hat{\delta}_{rot1})$$
$$\theta' = \theta + \hat{\delta}_{rot1} + \hat{\delta}_{rot2}$$

$$return\ x_t = (x', y', \theta')^T$$

Fig. 5. Odometry Motion Model Algorithm

B. *Measurement Model*

The measurement model $P(z_t \mid x_t)$ in the particle filter algorithm is obtained by processing the information from the omnidirectional vision sensor that captures 360° image as shown in Fig. 6. The value of this measurement model is used as a weghting of each particle.

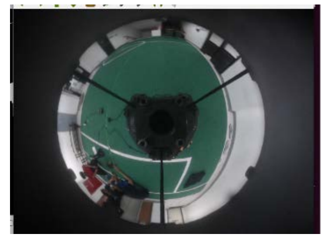


Fig. 6. Panoramic Image Captured by Omni-Vision

In the RoboCup MSL only the white lines of the field can be utilized as a measurement information source for localization process. So the radial scan lines method is applied to the image that has been calibrated for detecting the lines as shown in Fig. 7.

If n line points are detected, the relative coordinates of the detected line points to the robot for each particle can be defined as $f_i = \left(o_i^{[x]}, o_i^{[y]}\right)$, with $i = 1, 2, ..., \mathrm{n}$. The term $P\left(f_i \mid \chi_t\right)$ is the probability of detecting f_i when the robot is at $\chi_t = \left(x_t, y_t, \theta_t\right)$. The position for each f_i point in global coordinate can be determined by performing a geometry transformation according to (5).

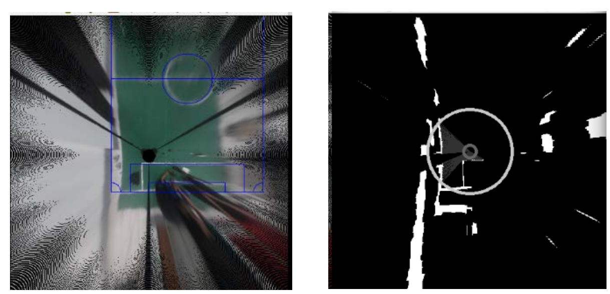


Fig. 7. Calibrated Image (Left) and Radial Scan Lines (Right)

$$o_i^{[m]} = \begin{pmatrix} x_t \\ y_t \end{pmatrix} + \begin{pmatrix} \cos\theta_t & -\sin\theta_t \\ \sin\theta_t & \cos\theta_t \end{pmatrix} \begin{pmatrix} o_i^{[x]} \\ o_i^{[y]} \end{pmatrix} \tag{5}$$

$P(f_i|\chi_t)$ can be determined by the probability of how f_i belongs to white mark lines corresponds to

$$P\left(f_i \mid \chi_t\right) = \exp\left(\frac{-d(o_i)d(o_i)}{2\sigma^2}\right) \tag{6}$$

Where σ is a constant. Therefore $P\left(f_i \mid \chi_t\right)$ can be calculated by the deviation between the distribution of o_i and the actual position of these points on the field, and $P\left(f_i \mid \chi_t\right)$ decreases as the deviation increases to the closest line named as $d(o_i)$. So the deviation only depends on o_i, and it can be precalculated and stored in two-dimentional look-up table. Fig. 8 shows $d(o_i)$ distribution on the field. From Fig. 8, the brightness represents the deviation scale, and the higher brightness depicts the smaller deviation, so we can obviously investigate that how the deviation change with varying o_i on the field.

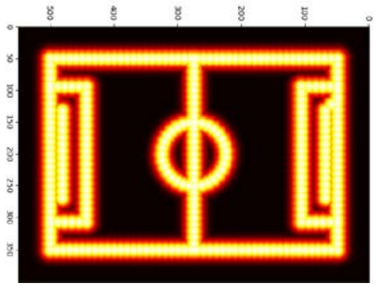


Fig. 8. The Distribution of $d(\boldsymbol{o_i})$ on The Field

Because $f_1, \ldots, f_n$ are detected independently, the sensor measurement model of the system can be represented as (7).

$$P(z_t \mid x_t) = P(o \mid \chi_t) = P(f_1 \mid \chi_t) \ldots P(f_n \mid \chi_t) \quad (7)$$

C. Resampling

The distribution of particles tends to degenerate where there particles with low weight due to dispersion in the prediction process. The selection / resampling stage is the key stage in the particle filter algorithm. The particle selection process needs to be carried out to keep the particle distribution in the correct posteriori distribution area.

Low Variance Sampler algorithm also called the systematic method is used for the resampling process. The algorithm is shown in Fig. 9. The probability for each particle $x_t^{[m]}$ to be resampled is proportional to its weight $w_t^{[m]}$. This algorithm converts the set of prior particles χ_{t-1} into a set of posterior particles χ_t and then rearranges the weight of the m-th particle to the same size $w_t^{[m]} = \frac{1}{M}$, with M is the number of particles **.**

The low-variance sampler is relatively easier to implement and has a computational complexity $O(M)$ which is faster than the independent selection method with complexity $O(M \log M)$ [9]. After the selection process is finished, the program will run recursively until the program is stopped.

Algorithm Low_variance_sampler($\boldsymbol{\chi_t}, \boldsymbol{W_t}$):

$\bar{\chi}_t = \emptyset$
$r = rand(0; M^{-1})$
$c = w_t^{[1]}$
$i = 1$
$for\ m = 1\ to\ M\ do$
$U = r + (m-1) \cdot M^{-1}$
$while\ U > c$
$i = i + 1$
$c = c + w_t^{[i]}$
$endwhile$
$add\ x_t^{[i]}\ to\ \bar{\chi}_t$
$endfor$
$return\ \bar{\chi}_t$

Fig. 9. The Low Variance Sampler Algorithm

V. Experimental Results

A. Sensor Characteristics

Compass

We use CMPS12 as the absolute orientaion sensor. By giving an angle of 360° into the compass for 100 times testing, we can obtain the sensor characteristics as shown in the Fig. 10.

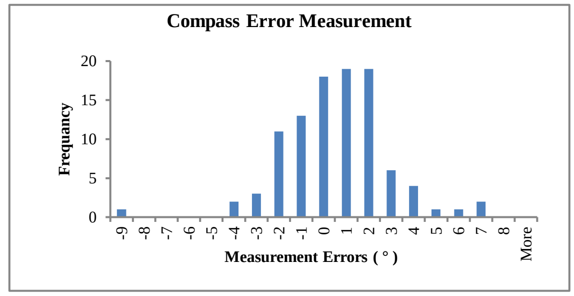


Fig. 10. Compass Measurement Errors Distribution

Normality test is carried out by using the *Kolmogorov – Smirnov* method at a significance level of 5%. From this test we can concluded that empirical data came from normally distributed populations as shown in Table I.

TABLE I.
Compass Measurement Errors Characteristic

Parameter	**Value**
Standar Deviation (σ)	2.31405°
Measurement Model	$\theta_{cmps} = \theta_{cmps} + \sigma.N(0,1)$

Encoders

To obtain the encoders sensor characteristic when used as an odometry, the robot is moved in the translational direction along one meter, two meter, and three meter for 100 times testing. The measurements is recorded and plotted as shown in Fig. 11. The measurement errors is calculated and sensor characteristic value is obtained as indicated by Table II.

TABLE II.
Compass Measurement Errors Characteristic

Parameter	**1 meter**	**2 meter**	**3 meter**
Mean	0.02	0.06	0.05
Sttdev (Y)	0.012104	0.091311	0.173108

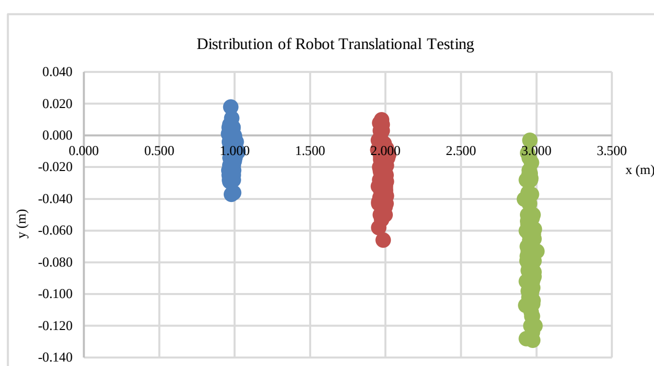


Fig. 11. Encoders Measurement Errors Distribution

B. *Simulation Results*

Odometry Motion Model

The simulation of odometry motion model is aimed to obtain the best noise parameter value in odometry motion model algorithm. By using 100 set of particle test that represent 100 times of testing and varying the values of the noise parameters from 0.00 until 0.1 we can obtain a plot of data as shown in Fig. 12.

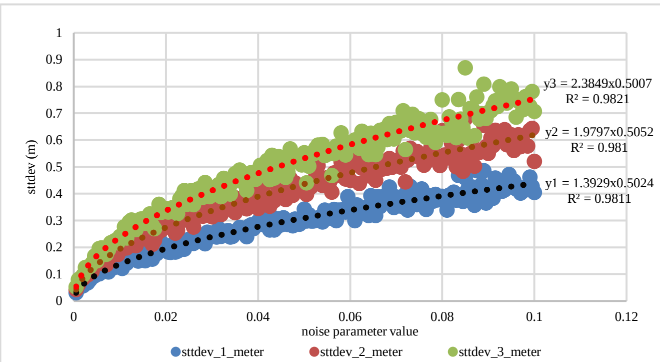


Fig. 12. Effect of Noise Parameters Against Distribution of Standard Deviation Values

Optimal noise parameter values can be obtained by performing optimization techniques on the objective function

$$F_{err} = |y1 - Y1| + |y2 - Y2| + |y3 - Y3| \quad (8)$$

where $y1$, $y2$, and $y3$ are measurement standar deviation from simulation models, and $Y1$, $Y2$, and $Y3$ are measurement standar deviation from the sensor realtime testing that we obtain before. From the calculation, the optimal noise parameter value is described as shown in Table III.

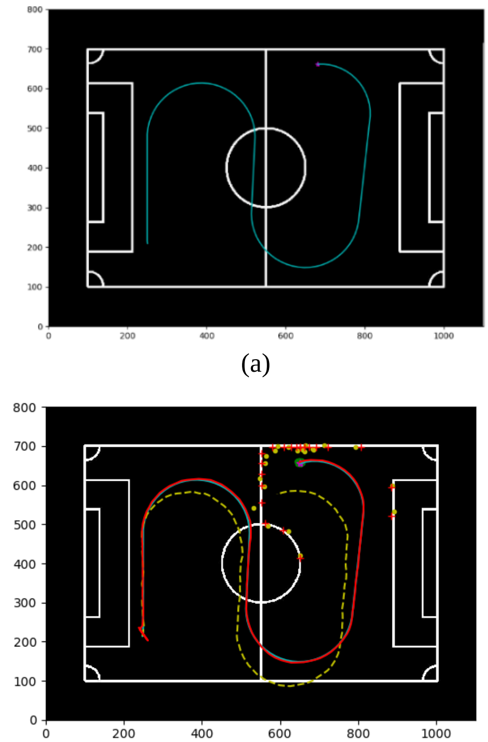
TABLE III.
NOISE PARAMETER OF ODOMETRY MODEL VALUE

Param	α_1	α_2	α_3	α_4	α_5
Value	0.0026	0.0026	0.0026	0.0026	0.0026

Particle Filter Simulation

Particle filter simulation is carried out to determine the optimum particle filter parameters before being tested on the robot. Particle filter algorithm is performed on the S-shaped path as shown with a blue line in Fig. 13 (a) to simulate changes in velocity and direction. The result of Particle filter simulation is shown in Fig. 13(b). Red line defines the esimate posisiton using filter particle, yellow line defines the estimate position using odometry and yellow dot points describe the detected line points with noise.

The performance of particle filters is determined by the correctness of the estimated results and the computational time required. Fig. 14(a) shows the plot of the particle filter estimation errrors in the simulation varying by number of particles. The RMS (Root Mean Square) error shows how fast the filter estimation reaches convergence to the actual position. Fig. 14(b) shows the computational time requaired corresponding to he number of particles.



(a)

(b)

Fig. 13. Simulation of Localization Using Particle Filter (a) Path or Actual Robot Motion (b) Simulation result

From Fig. 14, we can conclude that the greater the number of particles used, the estimated error value will be smaller but requires a longer computational time. From the data we can derive equation model for the estimation error and the computational time by (9) and (10).

$$err_{rms} = 933.11e^{0.01M} \quad (9)$$

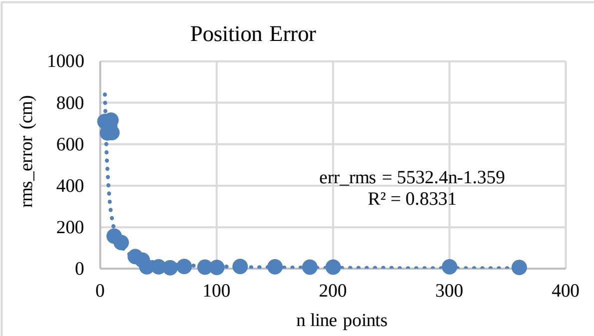
$$t = 0.0001M - 0.0013 \quad (10)$$

To obtain the optimal value of the particle number, the optimization technique is carried out on the objective function describe in (11).

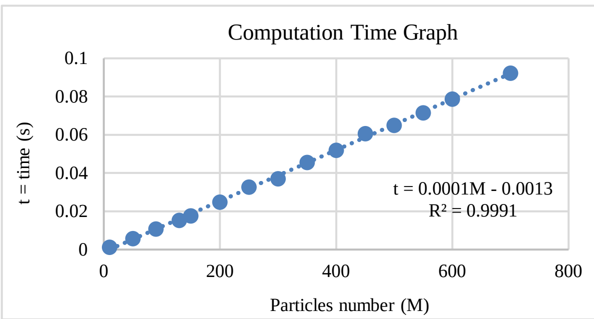
$$F_M = |err_{rms}| + |vt| \quad (11)$$

Where M is the particles number and v is the velocity given to the robot in the simulation. From the experiment, the optimal number of particles that can be used in the algorithm is ***M*=645** particles.

From Fig. 15, we can conclude that the computational time of the Particle Filter is dominated by weighting step. As we can see in (7), the weighting step depends on the number of n line points that being used in the algorithm. Graph of Particle Filter performance to the change of the number of n line points is shown in Fig. 16.



(a)



(b)

Fig. 14. Estimation Model on Varying Number of Particles (a) Position Measurement error (b) Computation Time

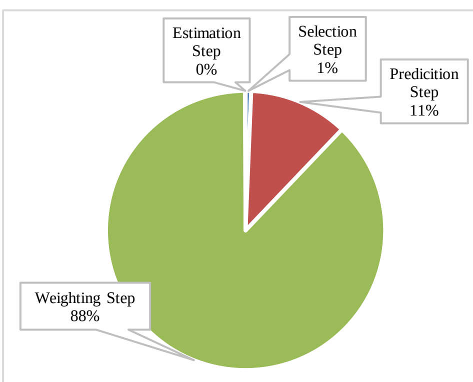


Fig. 15. Proportion of computation time for particle filter algorithms

From the figure above we can conclude that the greater the number of line points, the estimated error value will be smaller according to (12), but requires a longer computational time as described by (13).

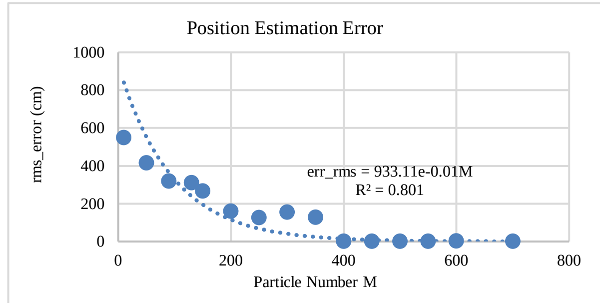
$$err_{rms} = 5532.4n^{-1.359} \quad (12)$$

$$t = 0.0026n + 0.0227 \quad (13)$$

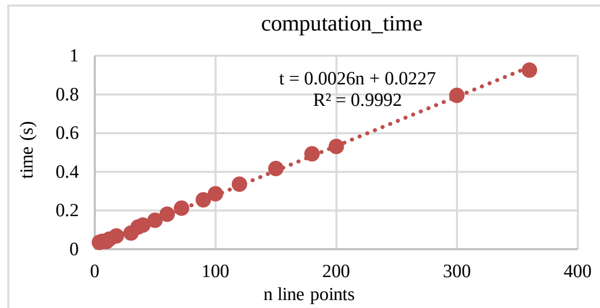
To obtain the optimal value of the line points number, the optimization technique is carried out on the objective function describe in (14).

$$F_n = |err_{rms}| + |vt| \quad (14)$$

From the experiment, the optimal number of line points that can be used in the algorithm is $\boldsymbol{n}$**=60** line points.



(a)



(b)

Fig. 16. Estimation Model on Varying Number Line Points (a) Position Measurement error (b) Computational Time

From the simulation, by using parameter M=645 particles and n=60 points, a comparison of localization error simulation using odometry and filter particle is shown by Table IV.

TABLE IV.
ESTIMATION ERROR OF ROBOT LOCALIZATION

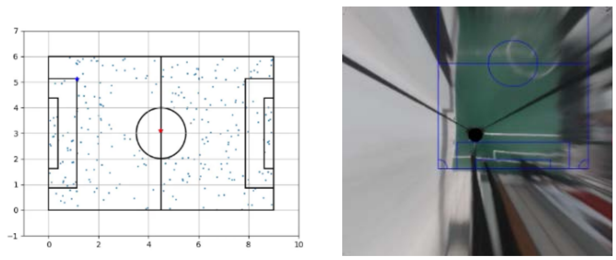
Method	Test Type	Position Error (cm)	Orientation Error
LLS /Odometry	RMS	64.488	0.119°
	Final	110.477	11.923°
Particle Filter	RMS	3.932	0.014°
	Final	2.941	0.39°

Realtime Implementation

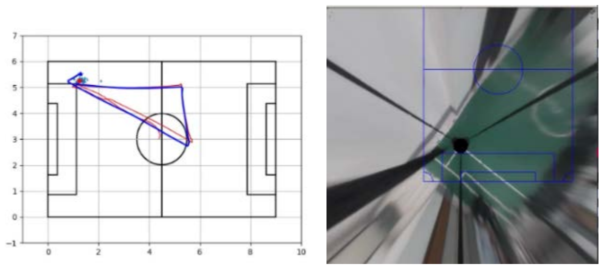
Particle Filter algorithm has been implemented in the robot by using the optimal parameters from simulation. The self-localization using Particle filter starts with initializes the particles position over the field uniformly, and uses data from the compass as the orientation initialization to eliminate ambiguity problem.

The experiement is carried out by moving the robot in the field on free paths and with accelerated motion upto maximum velocity 1.5m/s. The robot is driven through the control of the GUI system from basestation.

Fig. 17(a) shows the initial position and Fig. 17(b) shows the final position. Where the blue line shows historical position estimation data using odometry, while the red line is a historical position estimation data using Particle Filter, and green dots are the particles.



(a)



(b)

Fig.17. Realtime Implementation of self-Localization using Particle Filter (a) initial position and (b) final position

From Fig. 17, it can be seen that the Particle Filter algorithm can be implemented on realtime robot localization system. The final robot position in the experiment shows that Particle Filter has better robustness compared to odometry localization system.

VI. Conclusion

The self-localization system using Particle filter can be implemented on MSL mobile soccer robot by using encoders, compass, and omnidirectional vision sensor. The algorithm has optimal parameter M = 645 particles and n = 60 points. Localization system using Particle Filter has better performance than using LLS (odometry) method with estimated simulation error value of Particle filter of 3 cm while on the LLS of 110 cm.

VII. Acknowledgment

The authors would like to thank the Institut Teknologi Bandung Engineering Physic Department, the Instrumentation and Control Research Group, the Autonomous Vehicle Research Group, and the Dagozilla Mobile Soccer Robotics Team for the help and support so this paper could be finished.

References

[1] Robocup Federation, "ROBOCUP," Robocup Federation, 1 Agustus 2016. [Online]. Available: https://www.robocup.org/. [Accessed 2 Juli 2019].

[2] J. Borenstein, H. R. Everett, L. Feng and D. Wehe, "Mobile Robot Positioning: Sensors and Techniques," *Journal of Robotic Systems,* pp. 231-249, 1997.

[3] J. Borenstein and L. Feng, "Measurement and Correction of Systematic Odometry Errors in Mobile Robots," *IEEE Transactions on Robotics and Automation,* pp. 869 - 880, 1996.

[4] J. Inthiam and C. Deelertpaiboon, "Self-Localization and Navigation of Holonomic Mobile Robot using Omni-Directional Wheel Odometry," in *TENCON 2014 - 2014 IEEE Region 10 Conference*, Bangkok, 2014.

[5] H. Lu, X. li, H. Zhang, M. Hu and Z. Zheng, "Robust and Real-time Self-Localization Based on Omnidirectional Vision for Soccer Robots," *Advanced Robotics* , pp. 1-19, 2013.

[6] H. R. Moballegh, P. Amini, Y. Pakzad, M. Hashemi and M. Narmiani, "An Improvement of Self-Localization for Omnidirectional Mobile Robots Using A New Odometry Sensor and Omnidirectional Vision," in *Canadian Conference on Electrical and Computer Engineering* , Canada, 2004.

[7] H. P. Oliveira, A. J. Sousa, A. P. Moreira and P. J. Costa, "Modeling and Assesing of Omni-directional Robotswith Three and Four Wheels," in *IEEE International Conference on Robotics and Automation 1:521 - 527 vol.1*, Portugal, 2003.

[8] C. S. Putra, F. Fahleraz, A. Widyotriatmo and K. Mutijarsa, "Multilayer Control for Coordinating Three - Wheeled Omnidirectional Mobile Robots," in *2019 6th International Conference on Instrumentation, Control, and Automation (ICA)*, Bandung, 2019.

[9] S. Thrun, W. Burgard and D. Fox , Probablistic Robotics, Cambridge: MIT Press, 2005.

[10] C. Randell, C. Djiallis and H. Muller, "Personal Position Measurement Using Dead Reckoning," in *Seventh IEEE International Symposium on Wearable Computers,* New York, 2003.

Ahmad Wahrudin received his B.Eng. in engineering physics from the Institut Teknologi Bandung, Indonesia in 2019.

Since 2016, he was a research fellow with the Autonomous Vehicle Researh Group, Institut Teknologi Bandung, and member of ITB Dagozilla Mobile Soccer Robotics Team. His research interests include robotics, autonomous vehicles, automation system, and data analytic.

Augie Widyotriatmo (M'08) received his B.Eng. in engineering physics and M. Eng. in instrumentation and control from the Institut Teknologi Bandung, Indonesia in 2002 and 2006, respectively, and Ph.D. degree in mechanical engineering from Pusan National University, South Korea in 2011.

He joined the Faculty of Industrial Technology as a faculty member for engineering physics program in the Institut Teknologi Bandung Indonesia in 2010. He received honors of contribution in innovations from Institut Teknologi Bandung in 2016 for his invention on robotic wheelchair and in 2018 on cockroach robots. He also received the outstanding paper award in the 18th International Conference on Control, Automation and Systems (ICCAS 2018), South Korea, best paper awards in the 4th International Conference on Industrial Internet of Things 2018 (ICIIOT 2018), Indonesia, and in the 3rd International Conference on Robotics, Biomimetics, and Intelligent Computational Systems 2018 (ROBIONETICS 2018), Indonesia.

He was a visiting scholar in the University of Melbourne, Australia, in 2016. Dr. Widyotriatmo was the chair of the IEEE Indonesia Control Systems and Robotics and Automation Joint Chapter Societies for the period of 2017-2018. He is an associate editor in the International Journal of Control, Automation, and Systems (IJCAS). His research interests include robotics, autonomous vehicles, nonlinear control, energy optimization, biomedical instrumentation, automation system, and power system control.

Endra Joelianto (M'07--SM'19) received his B.Eng. in engineering physics from the Institut Teknologi Bandung, Indonesia in 1990, and his Ph.D. degree in Engineering from The Australian National University (ANU), Australia in 2002.

He has been with the Engineering Physics Study Program, Institut Teknologi Bandung, Indonesia as faculty member, since 1999. He has been a senior research fellow at the Centre for Unmanned System Studies (CENTRUMS), Bandung Institute of Technology, Bandung, Indonesia since 2007. He was a visiting scholar in the Telecommunication and Information Technology Institute (TITR), University of Wollongong, NSW, Australia in February 2002. He was a Visiting Scholar in the Smart Robot Center, Department of Aerospace and Information Engineering, Konkuk University, Seoul, Korea in October 2010.

He has edited one book on intelligent unmanned systems (Springer, 2009), published one book on Linear Quadratic Control: A State Space Approach (ITB Press, 2017) and published more than 125 research papers. Dr. Joelianto is a Senior Member IEEE. He was the chair of the IEEE Indonesia Control

Systems/Robotics Automation Joint Chapter Chair (2012-2016). He is the Chairman of Society of Automation, Control and Instrumentation, Indonesia. He is an Editor of the International Journal of Artificial Intelligence (IJAI) and the International Journal of Intelligent Unmanned Systems (IJIUS). He was the Guest Editor at the International Journal of Artificial Intelligence (IJAI), the International Journal of Imaging and Robotics (IJIR) and the International Journal of Applied Mathematics and Statistics (IJAMAS). His research interest includes hybrid control systems, discrete event systems, computational intelligence, robust control, unmanned systems and intelligent automation.