A New Odometry System to Reduce Asymmetric Errors for Omnidirectional Mobile Robots

Alireza Fadaei Tehrani[1,2], Ali Mohammad Doosthosseini[3], Hamid Reza Moballegh[1,3], Peiman Amini[1,3], and Mohammad Mehdi DaneshPanah[1,3]

[1] Robotic Center of Isfahan University of Technology
[2] Mechanical Engineering Department, Isfahan University of Technology (IUT)
[3] Electrical Engineering Department, Isfahan University of Technology (IUT)

Abstract. This work describes an investigation to reduce positioning error of 3 wheel middle size robot by using a modified odometry system. In this technique the positioning sensor (shaft encoders) are mounted on 3 free-running wheels so the slippage of the driving wheels does not affect the measurements of the sensors. This will result in decreasing the cumulative error of the system. This mechanism accompanying by omnidirectional vision system presents reliable and accurate self-localization method for any 3 wheel driving robot. Experimental results have shown performance improvement up to 86% in orientation error and 80% in position error.

1 Introduction

Self-localization is one of the most important issues in mobile robots. Different methods have been suggested for self-localization, naming a few, vision based self-localization, laser range finders, odometry [1], ultrasonic approach, gyroscope [5] and global positioning system (GPS). These methods mainly differ in their robustness, measuring accuracy, speed, cost and ease of implementation.

In odometry, robot displacement and change of direction are measured comparing to previous position. This method has acceptable results on platforms where robot moves on a smooth surface. Ease of implementation, relatively cheap and light weight instrument are some advantages of this method.

One of the platforms in which robots need self-localization is RoboCup middle size league (MSL). In this environment, because of predefined colors for different objects and smooth surface, vision based self-localization and odometry are assumed to be effective [6]. Laser range finders were also a good solution until the field walls were not removed [9]. Using vision based self-localization alone has some drawbacks:

Field landmarks are limited to flags, goals and border lines. Because of moving objects in the field, a few land marks may not be seen by robot. In addition, errors in object detection and the nonlinear map (from image units (pixels) to physical units (centimeters)) cause unreliable output. Robust image processing algorithms can enhance the precision, but the algorithms are complex and time consuming.

Adding another sensor and using sensor data fusion is assumed to be a suitable method for precision enhancement, odometry sensors are an example [1]. In case where sufficient landmarks can not be detected, odometry results would be helpful.

D. Polani et al. (Eds.): RoboCup 2003, LNAI 3020, pp. 600–610, 2004.

Variation in odometry results is smooth, this can help to detect and avoid sudden faulty reports of vision based self-localization. In the other hand, odometry errors are cumulative and regarding slippage, sensor outputs could be of low confidence.

In this article, three improvements are introduced for decreasing slippage error and enhancing sensor precision. The designed mechanical structure is also presented. Then two methods for position estimation are introduced and discussed in detail. Finally the implementation of a linear controller is described. The aim of this work is to implement a reliable navigation system, using modified odometry accompanying omnidirectional vision based self-localization.

2 Modified Odometry Sensors

Asymmetric errors are those errors which are caused by slippage and field unevenness. In MSL RoboCup, due to the flat playground, the highest portion of asymmetric error in odometry is caused by slippage when the robot accelerates or deaccelerates.

The designed structure includes an omni-directional system in which three omni-wheels are placed under the robot $120°$ apart.

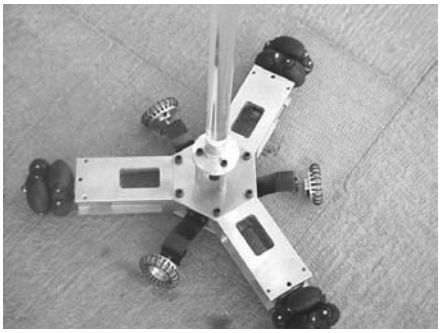


Fig. 1.

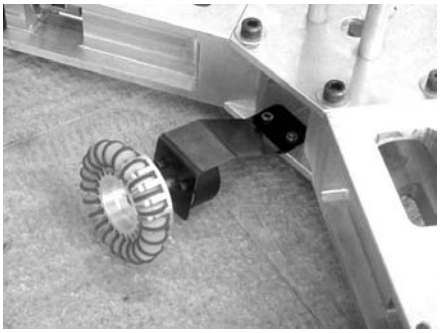


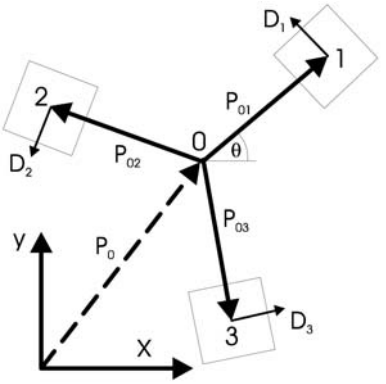
Fig. 2.

These free wheels are then coupled with shaft encoders and are assembled on a flexible structure, which ensures their firm contact with ground. The symmetry point of this structure coincides with the robot's center of mass and the wheels have an angular distance of $60°$ from the active (driving) wheels (Fig. 1,2). Most slippage occurs on the driving wheels during acceleration and changing direction.

Separating the sensors from driving wheels, decreases the slippage of the sensor wheels to a great extent, which results in decreasing the cumulative error of the system. In addition, special designed multi-roller wheels have vibration amplitude of less than 0.2 mm, also slippage decreases notably on carpet playground.

3 Robot Kinematics Model [3]

P_o is defined as the vector representing robot center of mass in (Fig.3). The position and tangential vector of each odometry wheel in the frame fixed to the robot are:



$$P_{o1} = L\begin{bmatrix}1\\0\end{bmatrix} \quad , \quad P_{o2} = R(\frac{2\pi}{3})P_{o1} \quad , \quad P_{o3} = R(\frac{4\pi}{3})P_{o1} \tag{1}$$

$$\mathrm{D}_1 = \begin{bmatrix}0\\1\end{bmatrix} \quad , \quad \mathrm{D}_2 = -\frac{1}{2}\begin{bmatrix}\sqrt{3}\\1\end{bmatrix} \quad , \quad \mathrm{D}_3 = \frac{1}{2}\begin{bmatrix}\sqrt{3}\\-1\end{bmatrix} \tag{2}$$

Fig. 3.

where L is the distance of wheels from the center of mass.

Using the above notations, the wheel position and velocity vectors can be expressed with the rotation matrix *R(θ)* by the following equations:

$$R_i = P_o + R(\theta)P_{oi} \tag{3}$$

$$V_i = P_o + \dot{R}(\theta)\dot{P}_{oi} \tag{4}$$

The angular velocity of each wheel can be expressed as:

$$\dot{\varphi}_i = \frac{1}{r}V_i^T(R(\theta)D_i) \tag{5}$$

where $\dot{\varphi}_i$ is the angular velocity and r is the radius of odometry wheels.

Substituting V_i from (4) into (5) yields:

$$\dot{\varphi}_i = \frac{1}{r}[\mathbf{P}_o^T R(\theta)D_i + \mathbf{P}_{oi}^T \dot{R}^T(\theta)R(\theta)D_i] \tag{6}$$

where the second term in the right hand side is the tangential velocity of the wheel. This tangential velocity could be also written as $L\dot{\theta}$, so from (6), we have:

$$L\dot{\theta} = \mathbf{p}_{Oi}^T \dot{R}^T(\theta)R(\theta)D_i \tag{7}$$

From the kinematics model of the robot, it's clear that the wheel velocities are linear functions of the robot linear and angular velocities.

$$\begin{bmatrix}\dot{\varphi}_1\\ \dot{\varphi}_2\\ \dot{\varphi}_3\end{bmatrix}=\frac{1}{r}\begin{bmatrix}-\sin\theta & \cos\theta & L\\ -\sin(\frac{\pi}{3}-\theta) & -\cos(\frac{\pi}{3}-\theta) & L\\ \sin(\frac{\pi}{3}+\theta) & -\cos(\frac{\pi}{3}+\theta) & L\end{bmatrix}\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{bmatrix} \qquad \dot{\Phi}=\frac{1}{r}W.\dot{S} \tag{8}$$

4 Self-localization Using Odometry

In this section two self-localization methods based on shaft encoder outputs are introduced, which are called differential method and direct method.

4.1 Differential Method

In this method, the outputs of shaft encoders are differentiated temporally resulting the individual wheel velocities. Using the kinematics equations, the robot linear and angular velocities are computed. Integrating these velocities, one can extract the robot position. The algorithm for this method can be described as fallows:

Using kinematics equation of the robot (8), x , y and θ are the parameters to be computed by having $\dot{\varphi}_i$ from shaft encoders. Solving equation (8) using the inverse of W results:

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{bmatrix}=r\begin{bmatrix}\frac{\cos(\pi/3+\theta)-\cos(\pi/3+\theta)}{3\sin\pi/3} & \frac{-\cos\theta-\cos(\pi/3+\theta)}{3\sin\pi/3} & \frac{\cos\theta+\cos(\pi/3-\theta)}{3\sin\pi/3}\\ \frac{\sin(\pi/3+\theta)+\sin(\pi/3+\theta)}{3\sin\pi/3} & \frac{-\sin\theta-\sin(\pi/3+\theta)}{3\sin\pi/3} & \frac{\sin\theta+\sin(\pi/3-\theta)}{3\sin\pi/3}\\ \frac{1}{3L} & \frac{1}{3L} & \frac{1}{3L}\end{bmatrix}\begin{bmatrix}\dot{\varphi}_1\\ \dot{\varphi}_2\\ \dot{\varphi}_3\end{bmatrix}$$

$$\dot{S}=rW^{-1}.\dot{\Phi} \tag{9}$$

where $\dot{\varphi}_i$ can be calculated from the current encoder samples, compared with previous samples. Using numerical integration methods on $\begin{bmatrix}\dot{x} & \dot{y} & \dot{\theta}\end{bmatrix}^T$ (knowing the initial position of the robot), the new position can be obtained.

Although this method is very simple, but due to inherent differentiation operations, it is not reliable in long term. The main drawback of differential method is the error accumulation. Some of important errors are: error in measuring time intervals, errors caused by the quantized nature of shaft encoders and output errors due to floating point computations.

Practically, time is divided into equal intervals, ΔT , in which the robot wheels' velocities are assumed to be constant. Considering these constant speeds, both robot

linear and angular velocities are determined, and the head angle and robot position would be grown with the extracted differential values.

The assumption of constant velocities in each time interval ΔT, will cause some error, which in turn will decrease the reliability of the computation results as time passes. In addition, limited precision of shaft encoders will result in some kind of quantization error which therefore leads to inaccuracy, the last point is that this error might increase due to digital computations. In order to decrease these errors, one possible solution is to define ΔT as small as possible and to increase shaft encoder resolutions. Needless to remark that this solution will increase the calculation's overhead.

4.2 Direct Method

The differential method was described as digitally differentiating the output of shaft encoders, applying the kinematics equations of the robot to the result of differentiation and integrating over the results.

Because of quantized nature of the values in time, integration and differentiation can not be implemented without approximation. If these operations could be eliminated and results can be driven directly from the measured angle by shaft encoders, more accuracy in calculation results are then achieved.

Using the equations of motion (9), in order to calculate x, y, θ directly, the following integral should be computed:

$$\dot{S} = r\int W^{-1}.\dot{\Phi}\ dt \tag{10}$$

where

$$S = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} = \begin{bmatrix} x(\varphi_1, \varphi_2, \varphi_3) \\ y(\varphi_1, \varphi_2, \varphi_3) \\ \theta(\varphi_1, \varphi_2, \varphi_3) \end{bmatrix} \tag{11}$$

system of equations in (10), results:

$$\mathrm{x} = r\int \frac{1}{3\sin(\frac{\pi}{3})} \left[(\cos(\frac{\pi}{3}+\theta) - \cos(\frac{\pi}{3}-\theta))\,\dot{\varphi}_1 + (-\cos(\theta\mathrm{o} - \cos(\frac{\pi}{3}+\theta))\,\dot{\varphi}_2 + (\cos(\theta\mathrm{c} - \cos(\frac{\pi}{3}-\theta))\,\dot{\varphi}_3 \right] \mathrm{d}t \tag{12}$$

$$\mathrm{y} = r\int \frac{1}{3\sin(\frac{\pi}{3})} \left[(\sin(\frac{\pi}{3}-\theta) - \sin(\frac{\pi}{3}+\theta))\,\dot{\varphi}_1 + (-\sin(\theta\mathrm{i} - \sin(\frac{\pi}{3}+\theta))\,\dot{\varphi}_2 + (\sin(\theta\mathrm{s} - \sin(\frac{\pi}{3}-\theta))\,\dot{\varphi}_3 \right] \mathrm{d}t \tag{13}$$

$$\theta = r\int \frac{\dot{\varphi}_1 + \dot{\varphi}_2 + \dot{\varphi}_3}{3\mathrm{L}}\,\mathrm{d}t \tag{14}$$

Further simplification of (14) results:

$$\theta = \frac{r}{3\mathrm{L}}\left[\int \dot{\varphi}_1\,\mathrm{d}t + \int \dot{\varphi}_2\,\mathrm{d}t + \int \dot{\varphi}_3\,\mathrm{d}t\right] = \frac{r}{3\mathrm{L}}(\varphi_1 + \varphi_2 + \varphi_3) \tag{15}$$

It's apparent from (15), that θ could be easily calculated from shaft encoder outputs, but it is impractical to extract x or y as direct functions of φ_1, φ_2, φ_3, from equations (12), (13). This could be demonstrated with the following example:

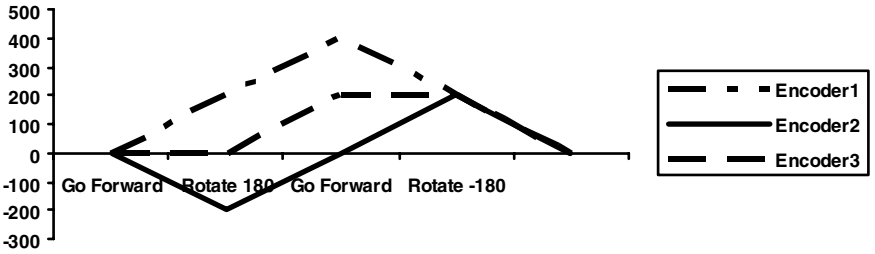


Fig. 4. The robot has the displacement of 2L however the shaft encoder outputs are the same as initial position.

In the above example, robot has started moving from an initial position. After moving a straight path with length L, it rotates 180° and then moves straight the same length along the previous direction and then rotates -180°. It is clear from (Fig.4) that for both initial and end points, $\varphi_1 = \varphi_2 = \varphi_3 = 0$, while the robot has a displacement of 2L. Therefore direct computation of x and y from $\varphi_1, \varphi_2, \varphi_3$ is not practical, which necessitates some restrictions on x and y to be extractable directly.

As it is understood from equations (12), (13) the coefficients of $\dot{\varphi}_i$ are functions of θ which itself is a function of time. Assuming θ to be independent of time, equations (12), (13) could be rewritten as follows:

$$x = \frac{r}{3\sin\frac{\pi}{3}}\left[\int\left(\cos\left(\frac{\pi}{3}+\theta\right)-\cos\left(\frac{\pi}{3}-\theta\right)\right)\dot{\varphi}_1\,dt+\int\left(-\cos\theta-\cos\left(\frac{\pi}{3}+\theta\right)\right)\dot{\varphi}_2\,dt+\int\left(\cos\theta+\cos\left(\frac{\pi}{3}-\theta\right)\right)\dot{\varphi}_3\,dt\right]$$

$$=\frac{r}{3\sin\frac{\pi}{3}}\left[\left(\cos\left(\frac{\pi}{3}+\theta\right)-\cos\left(\frac{\pi}{3}-\theta\right)\right)\varphi_1+\left(-\cos\theta-\cos\left(\frac{\pi}{3}+\theta\right)\right)\varphi_2+\left(\cos\theta+\cos\left(\frac{\pi}{3}-\theta\right)\right)\varphi_3\right] \tag{16}$$

$$y = \frac{r}{3\sin\frac{\pi}{3}}\left[\int\left(\sin\left(\frac{\pi}{3}-\theta\right)+\sin\left(\frac{\pi}{3}+\theta\right)\right)\dot{\varphi}_1\,dt+\int\left(-\sin\theta-\sin\left(\frac{\pi}{3}+\theta\right)\right)\dot{\varphi}_2\,dt+\int\left(\sin\theta-\sin\left(\frac{\pi}{3}-\theta\right)\right)\dot{\varphi}_3\,dt\right]$$

$$=\frac{r}{3\sin\frac{\pi}{3}}\left[\left(\sin\left(\frac{\pi}{3}-\theta\right)+\sin\left(\frac{\pi}{3}+\theta\right)\right)\varphi_1+\left(-\sin\theta-\sin\left(\frac{\pi}{3}+\theta\right)\right)\varphi_2+\left(\sin\theta-\sin\left(\frac{\pi}{3}-\theta\right)\right)\varphi_3\right] \tag{17}$$

Assuming θ to be independent of time, will result in $\dot{\theta}$ to be zero, thus:

$$\dot{\theta}=0\Rightarrow\frac{1}{3L}\left(\dot{\varphi}1+\dot{\varphi}2+\dot{\varphi}3\right)=0\Rightarrow\dot{\varphi}1+\dot{\varphi}2+\dot{\varphi}3=0 \tag{18}$$

Equation (18) shows that robot position can be extracted directly if the robot has had no rotation while moving to the new position. If $\dot{\theta}$ changes to a value other than zero, the origin of the coordinate system should be moved according to the calculated robot displacement vector and the computation should be started over again. In practice the condition on $\dot{\theta}$ is satisfied if the difference between the new head angle and the last stored angle exceeds some certain threshold.

Although the above equations are extracted assuming θ to be constant, for further improvement in precision, θ in (16,17) is replaced dynamically with the instantaneous angle which is easily computed from (15). In the method described above, the number of times which error accumulation takes place has been reduced to a great extent, this will result in improving the overall performance of direct method comparing to the differential method.

5 Robot Controller

In [10] it has been shown that two identical PID controllers for robot position and orientation are suitable for controlling a robot. If the residual error is negligible the integrator can be omitted. This controller has shown to be robust enough for controlling a soccer player robot. P and D coefficient are adjusted manually.

5.1 Position Controller Architecture

In order to implement the position controller, the position error vector is determined as follows:

$$\vec{e} = \begin{bmatrix} x \\ y \end{bmatrix} - \begin{bmatrix} x' \\ y' \end{bmatrix} \tag{19}$$

$$\vec{V} = k_d \frac{d\vec{e}}{dt} + k_p \vec{e} \tag{20}$$

where $\vec{V}$ expresses the output of the position controller for driving units, which it's components on each driving wheel are extracted with:

$$\mathrm{V_i} = \vec{\mathrm{V}}^{\mathrm{T}}.\mathrm{D_i} \tag{21}$$

5.2 Head Angle Controller

Assuming that the head angle of the robot is β and the desired head angle is θ.The angle controller is then determined as follows:

$$w = k_p e_\theta + k_d \frac{de_\theta}{dt} \tag{22}$$

where $e_\theta = \theta - \beta$ is the angle error of the robot.

5.3 Scaling

The output of the position or angle controller in Equations (21), (22) may exceed the range of the applicable voltages on the motors. In order to deal with such a situation, a scaling algorithm is believed to be efficient.

If the output vector of position controller was out of range of maximum motor voltage (v_{Max}), it is scaled down with the following scale factor:

$$\alpha = \frac{v_{Max}}{\max|V_i|} \tag{23}$$

For the angle controller, another scaling method is used, in which the output of the angle controller is clipped to $\beta v_{\max}$ ($0<\beta<1$). The reason for using such threshold is that a relatively high angle error would result in high motor voltages and this in turn affects the output of position controller.

5.4 The Final Motor Controller

The final applicable voltages on the motors are computed as:

$$u_i = v_i + w \tag{24}$$

The following scale factor is used for u_i before being applied to each motor if $\max|u_i| > u_{\max}$:

$$\gamma = \frac{u_{Max}}{\max|u_i|} \tag{25}$$

6 Experimental Results

In order to check the improvement of the odometry system introduced in this article compared to similar systems, a test condition was suggested to measure the position and head angle error.

In the designed test condition, care was taken to simulate the real situation. The test condition and the results are presented in the following two sections. At the end, the effect of the odometry sensor in reducing vision based self-localization overhead will be discussed.

6.1 Test Condition

UMBmark method is a popular benchmark for odometry systems [8], but the following test condition was used for simulating robot behavior more closely.

In this test, the robot is placed at one end of the field, and then the robot moves straight to the other end (10 meters) while rotating $360°$. The position and head

angle error are then measured. This procedure is repeated several times, and the mean values are computed.

In the following figure, shaft encoder outputs of an ideal odometry system under the test condition are presented.

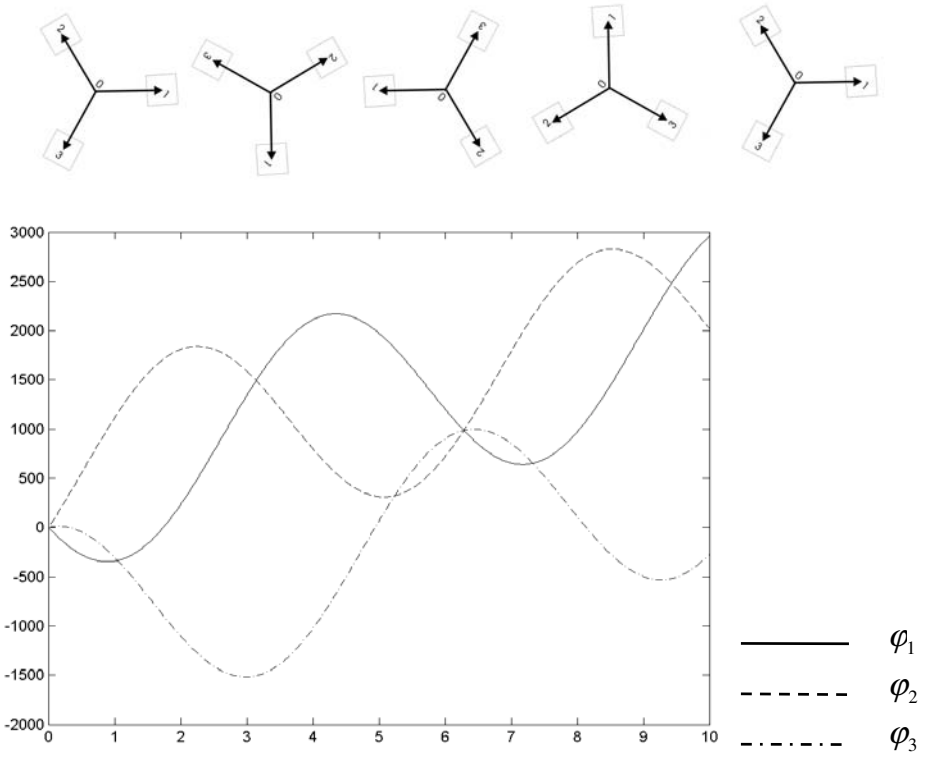


Fig. 5. Up: Position and orientation of robot while moving straight in 2.5 meters intervals. Down: The output of shaft encoders plotted over the length of 10 meters under the test condition.

6.2 Test Results

The following table demonstrates a comparison between robots with the ordinary odometry sensor, which the shaft encoders are connected to driving wheels, and the next generation robot in which the improved odometry system was installed on. The direct method was used for position calculations.

Using the improved sensors, the relative position error is 1.5%, while the relative orientation error is 1.1%. So, the new odometry sensors have improved the orientation error to 86%, and the position error to 80%. In this way we have reduced the asymmetric errors to a great extent.

Table 1. The absolute and relative position and angle error of ordinary and improved sensors.

	Position Error	Orientation Error	Relative Pos. Error	Relative Orientation Error
Ordinary sensors	75cm	30°	7.5%	8.2%
Improved sensors	15cm	4°	1.5%	1.1%

Considering the position accuracy, the vision based self-localization can be updated with longer time intervals, while the odometry results are used for self-localization. This in turn enables us to perform accurate, complicated and time consuming vision based self-localization algorithms.

Having relatively precise information about orientation is also used in searching algorithms of stationary objects (goals and flags), which causes the image processing algorithms more robust.

7 Conclusions

Joint visualization – odometry localization of moving robots has attracted the attention of many researchers. In this work we intended to improve the performance of the existing algorithms by proposing a new odometry system with lower asymmetric errors. In our approach there are free wheels accompanying the driving wheels solely to measure the robot rotation and displacement. Hence slippage is reduced considerably and error accumulation diminishes. New methods for converting measurements into desirable data are also proposed to avoid the amplification of inaccuracies in differentiations. Test results verified the improvements clearly.

References

1. Fredric Chenavier,James L. Crowley: "Position Estimation for a Mobile Robot Using Vision and Odometry", *Proceeding of IEEE International Conference on Robotics and Automation*
2. Johann Borenstein: "Experimental Results from Internal Odometry Error Correction with the OmniMate Mobile Robot ", *IEEE Transactions on Robotics and Automation, Vol. 14, NO. 6, December 1998 963*
3. Tamás Kalmár-Nagy, Rafaello D'Andrea, Pritam Ganguly: " Near-Optimal Dynamic Trajectory Generation and Control of an Omnidirectional Vehicle " ,*Cornell University April 8 2002*
4. Agostino Martinelli, " Possible Strategy to Evaluate the Odometry Error of A Mobile Robot ",*Proceeding of The 2001 IEEE/RSJ International Conference On Intelligent Robots and*
5. K. Komoriya, E. Oyama, "Position Estimation of a Mobile Robot Using Optical Fiber Gyroscope (OFG)", *Proc. Int. Conf. Intell. Robot. Syst. (IROS'94), Munich, Germany, pp.143-149, Sept. 12–16, 1994.*
6. Ashly W. Stroup,Kevin Sikorki,and Tuker Balch**:** "Constrained – Based Landmark Self-localization" ,*The 2002 International RoboCup Symposium*

7. Kok Seng Chong ,Lindsay Kleeman, " Accurate Odometry and Error Modeling for a Mobile Robot", Proceeding of the 1997 IEEE International Robotics and Automation
8. J. Borenstein and L. Feng, "UMBmark: A Benchmark Test for Measuring Dead-Reckoning Errors in Mobile Robots", *Proc. 1995 SPIE Conf. Mobile Robots.*
9. Hans Utz, Alexander Neubeck , Gerd Mayer, Gerhard Kraetzschmar. "Improving Vision-Based Self Localization" ,*The 2002 International RoboCup Symposium*
10. Keigo Watanabe, "Control of an Omnidirectional Mobile Robot", *Second International Conference Acknowledge-Based Intelligent Electronic Systems,21-23 April*