Proceedings of the 2005 IEEE/ASME
International Conference on Advanced Intelligent Mechatronics
Monterey, California, USA, 24-28 July, 2005

Feedforward Global/Inertial Sensor Fusion Algorithm for Accurate Global Positioning of a Mobile Robot

Kyoobin Lee, Jaeheung Park, Oussama Khatib and Dong-Soo Kwon, Member, IEEE

Abstract— **This paper introduces a coordinate transform method for global/inertial sensor fusion minimizing modification of an existing control program of a mobile robot. Most of GPS/INS sensor fusion algorithms use Kalman filters and modify the INS states by feedback loops. Because the structure of the proposed method has a feedforward filter, the proposed method has an advantage in case the user does not want to change an existing control program of mobile robot. The feedback type Kalman filter is designed so that the error between global position from GPS and odometry from INS converges to zero. Therefore the coordinate matching between the odometry and measured global position is not necessary in those approaches. However, in the feedforward structure described in this paper, the errors gradually increase over time. A coordinate transform method has been developed for dealing with the error. This method provides an easy way to make an add-on function without any changes in the existing functions of the control program of a mobile robot.**

I. INTRODUCTION

THE development of an autonomous navigation system, such as indoor service robot, requires high performance localization and position sensor technology. There are two types of position measurement devices, relative sensors and absolute sensors, and both are problematic. Encoders are widely used as relative position sensors for mobile robots. Service robots generally use global localization using GPS, laser range sensors, or ultrasonic sensors to determine absolute position. Relative position sensors usually have high resolution, fast response time, and low noise. However, they also suffer from unpredictable drift errors or systematic errors due to misaligning because relative sensors are equipped in the system. Systematic odometry errors have been well analyzed in [2, 3, 4]. On the other hand, although absolute position sensors, like the GPS or localization system with laser range sensors, have low resolution, slow response time, and large noise, they do not suffer from drift errors because they do not use an integration of incremental signals. In order to take advantages of sensor fusion of the relative and absolute sensors, many algorithms are developed. Kalman filters are widely used for the sensor fusion between the relative sensors and the absolute sensors [5,6,7].

Recently, there are many commercial products of mobile robots. The vendors also provide the source code of control functions or developing library toolkits to easily build the control program. However, because the conventional sensor fusion approaches usually use feedback structure, users have to modify the existing program in order to apply the sensor fusion algorithms. To minimize the software modification, feedforward filter can be used [8]. In this paper, a coordinate transform method is described to provide a convenient way adding sensor fusion algorithm to the existing system with minimum software modification. The proposed method is not for filter design but for coordinate matching between the existing control program and the added filter. Therefore, any types of filter can be used with the proposed method.

Section 2 defines the problem and terminology used. Detail explanation of the proposed coordinate transform method is described in section 3. Section 4 shows an example code of the sensor fusion. Section 5 describes the corresponding control method. In section 6, an experiment with a mobile robot is described. Finally, the conclusions are given in section 7

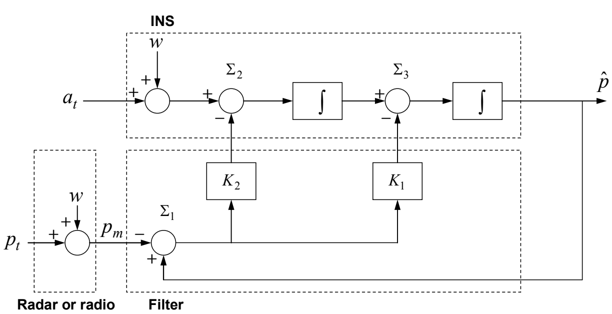

Fig. 1. Feedback structure. a_t : true acceleration. w : noise. $\hat{p}$: computed global position. p_t : true global position. p_m : measured global position. K_1 , K_2 : filter gain. [1]

II. PROBLEM STATEMENT

Filters used in GPS/INS sensor fusion is usually feedback structure as shown in figure 1 [1]. A simplest model of INS is used here because the proposed method is not for filter modeling. The INS model and filter can be chosen by user's will while using the proposed method. The error between measured global position, p_m , and computed position, $\hat{p}$, is

Manuscript received March 8, 2005. This work was supported by the International Research Internship Program of the Korea Science and Engineering Foundation (KOSEF), and by the Intelligent Robotics Development Program, one of the 21st Century Frontier R&D Programs funded by the Ministry of Commerce, Industry and Energy of Korea.

Dong-Soo Kwon and Kyoobin Lee are with Korea Advanced Institute of Science and Technology (KAIST), 305-701 Korea (corresponding author to provide phone: +82-42-869-3042; fax: +82-42-869-3210; e-mail: kwonds@kaist.ac.kr)

Oussama Khatib and Jaejeung Park are with Stanford University, CA 94305 USA (e-mail: khatib@cs.stanford.edu)

0-7803-9046-6/05/$20.00 ©2005 IEEE.

fed back into the states in the INS module by multiplying the filter gains, K_1 and K_2.

Consider the situation that the robot stops while moving. When the feedback gains are well defined, the error will converge to zero. If the feedback filter works appropriately, the measured global position, p_m, and the computed position, $\hat{p}$, are asymptotically on the same coordinate frame. Therefore, in this approach, the summing junctions, Σ_1, Σ_2 and Σ_3 can be just defined as vector sum even in multi degree-of-freedom motion.

The feedback filters have to modify the states of the INS. This unavoidably causes modifying the odometry generating functions that include reading the incremental amounts of encoder or acceleration, integrating them to the odometry according to the robot dynamics. If the mobile robot is developed by the user, those works are not difficult. However, if the robot is a commercial one and the provided control functions are encapsulated such as an executive library, then the users might feel uncomfortable to modify or re-make them. The proposed feedforward structure of sensor fusion is developed to solve this problem.

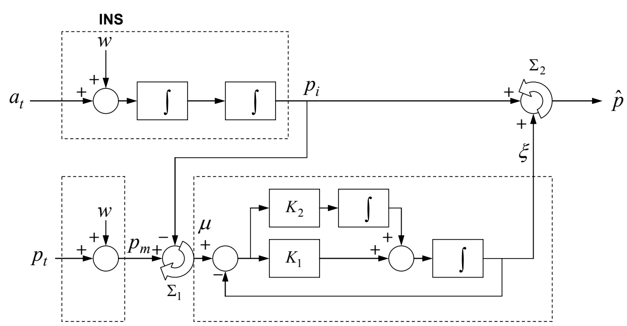

Fig. 2. Feedforward structure. a_t : true acceleration. w : noise. p_i : odometry. $\hat{r}$: computed global position. r_t : true global position. r_m : measured global position. K_1, K_2 : filter gain. [1]

The feedback structure is able to be equivalently converted to a feedforward structure as shown in figure 2 [1]. In that case, the input of the feedforward filter does not converge to zero because the odometry of INS always has drifting errors.

When we use this configuration for a one degree-of-freedom problem, the sensor fusion may work correctly. However, in multi degree-of-freedom problem, it cannot be used in the same way. In this paper, we consider a planar motion for controlling mobile robots. Consider again the situation that the robot stops while moving. After the filter output, ξ, converges to the error between the odometry and the measured global position, the computed global position, $\hat{p} = p_i + filter(p_m - p_i) = p_i + filter(\mu) = p_i + \xi$, converges to the measured global position, p_{m1} (See p_{m1}, $\hat{p}_1$, and p_{i1} in figure 3). However, when the robot start again, the $\hat{p}_2$ goes to wrong direction in comparison with the correct direction, p_{m2}. This is because the summing junctions are considered as vector sum. This problem can be solved by substituting appropriate coordinate transform for the summing junctions, Σ_1 and Σ_2.

In order to correct the problem, it is necessary to introduce trajectory transform in planar motion. The method described in figure 3 holds the start position of the p_i and p_m together at the same position. On the other hand, when we match the end position as shown in figure 4, the problem is eliminated because now the heading angles are equal. The filter input, μ, can be obtained by back-stepping along the trajectory of p_i from the end point of p_m. This is the key idea of the proposed algorithm.

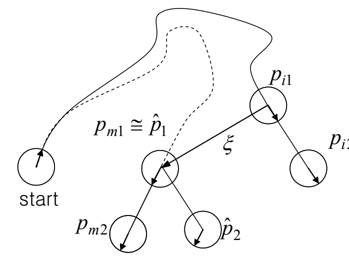

Fig. 3. An illustration assuming that the summing junction, Σ_1 and Σ_2 are vector sum

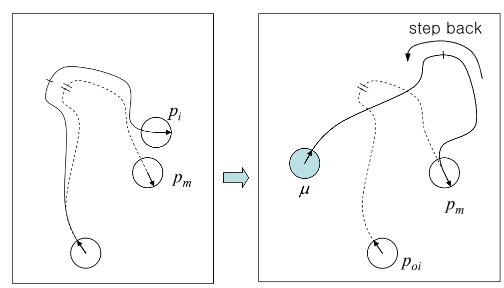

Fig. 4 At the summing junction, Σ_1, the trajectories of odometry and measured global position are considered.

III. Coordinate Transformation for Sensor Fusion

A. Definition of Trajectory Transform

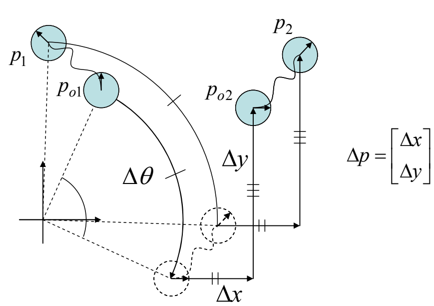

Fig. 5. Trajectory transform. The trajectory from p_{o1} to p_1 is identical to the trajectory from p_{o2} to p_2.

The trajectory transform is defined as follows. Consider a robot moves along a given trajectory from a starting position,

p_{o1}, and arrives at p_1 as shown in figure 5. If the robot moves along the same trajectory from another starting position, p_{o2}, then the arriving position will be on p_2. It can be derived as following. At first, the relationship between the starting positions is,

$$\begin{bmatrix} p_{o2p} \\ 1 \end{bmatrix} = \begin{bmatrix} R(\Delta\theta) & \Delta p \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_{o1p} \\ 1 \end{bmatrix}$$
$$p_{o2\theta} = p_{o1\theta} + \Delta\theta$$
$$\text{where } p = \begin{pmatrix} p_p \\ p_\theta \end{pmatrix} = \begin{pmatrix} p_x \\ p_y \\ p_\theta \end{pmatrix} \tag{1}$$
$$\text{the rotation matrix, } R(\Delta\theta) = \begin{bmatrix} \cos\Delta\theta & -\sin\Delta\theta \\ \sin\Delta\theta & \cos\Delta\theta \end{bmatrix}$$

Then Δx, Δy, and $\Delta\theta$ can be expressed with terms, p_{o1} and p_{o2} as following,

$$\begin{aligned} \Delta\theta &= p_{o2\theta} - p_{o1\theta} \\ \Delta p &= p_{o2p} - R(p_{o2\theta} - p_{o1\theta}) \cdot p_{o1p} \end{aligned} \tag{2}$$

The relationship between p_1 and p_2 is identical to the relationship between p_{o1} and p_{o2} because both of the traveling trajectories are identical. Therefore p_2 can be obtained by Eq. (1) with same $\Delta\theta$ and Δp.

$$\begin{bmatrix} p_{2p} \\ 1 \end{bmatrix} = \begin{bmatrix} R(p_{o2\theta} - p_{o1\theta}) & p_{o2p} - R(p_{o2\theta} - p_{o1\theta}) \cdot p_{o1p} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_{1p} \\ 1 \end{bmatrix} \tag{3}$$
$$p_{2\theta} = p_{1\theta} + p_{o2\theta} - p_{o1\theta}$$

In order to perform the position transform and the angle transform at the same time, the trajectory transformation from p_{o1} to p_{o2} is defined as following,

$$\begin{bmatrix} p_2 \\ 1 \end{bmatrix} = {}^{p_{o2}}_{p_{o1}}T \begin{bmatrix} p_1 \\ 1 \end{bmatrix}$$
$$\text{where } {}^{p_{o2}}_{p_{o1}}T = \begin{bmatrix} R(p_{o2\theta} - p_{o1\theta}) & 0 & p_{o2p} - R(p_{o2\theta} - p_{o1\theta}) \cdot p_{o1p} \\ 0 & 1 & p_{o2\theta} - p_{o1\theta} \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

B. Procedures for Sensor Fusion

The procedures to get the corrected global position are shown in figure 6. Detail explanations are as follows.

Procedure (a, b) — In practice, the starting positions of odometry and the starting position of global position are usually not equal (figure 6a.) The trajectory from the starting position of the measured global position can be converted using the trajectory transformation so that its starting position is moved to the starting position of odometry (figure 6b.)

$$\begin{bmatrix} p'_m \\ 1 \end{bmatrix} = {}^{p_{oi}}_{p_{om}}T \begin{bmatrix} p_m \\ 1 \end{bmatrix} \tag{5}$$

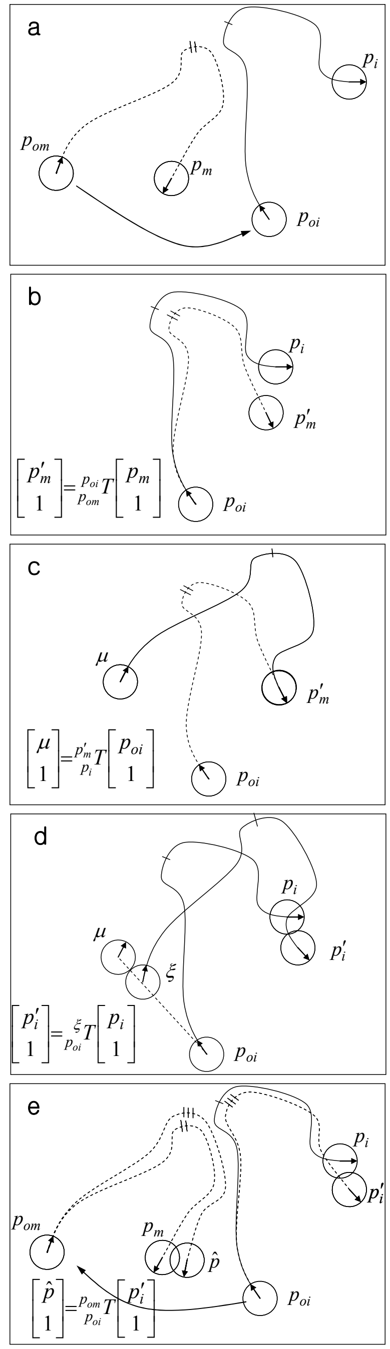

Fig. 6. The procedures of the proposed method. a, b, and c are for the junction, Σ_1 in figure 2. d and e are for the junction Σ_2

Procedure (c) — Obtaining the filter input, μ, in the figure 2 is more complicated. The starting position of odometry, p_{oi}, can be considered back-stepped trajectory from the end position of odometry, p_i. The starting position of p_{oi} is shifted from p_i to the measured global position,

$p_m'^2$, using the trajectory transform (figure 6c.) This is the backstepping concept described in section II (See figure 4).

$$\begin{bmatrix} \mu \\ 1 \end{bmatrix} = {}^{p_m'}_{p_i}T \begin{bmatrix} p_{oi} \\ 1 \end{bmatrix} \tag{6}$$

Those procedures, a, b and c, occur at the junction Σ_1 of the filter. The output, ξ, is obtained by filtering the μ. If the filter gains are determined as Kalman filter, the feedforward filter shown in figure 2 will be a second order low-pass filter which provides minimum settling time. It is because $K_1^2 / K_2 = 2$ [1]. The role of this low-pass filter is to smooth the measured global position. The measured global position from GPS or laser-localization system is usually coarse and has slow update rate. As stated above, the filter does not have to be a Kalman filter. Users can choose any type of filter that is able to make the measured global position smooth enough to use it for controlling the mobile robot.

Procedure (d) — At the summing junction, Σ_2, the corrected odometry, p_i', can be obtained by shifting the starting point of the odometry, p_i, to the filter output, ξ.

$$\begin{bmatrix} p_i' \\ 1 \end{bmatrix} = {}^{\xi}_{p_{oi}}T \begin{bmatrix} p_i \\ 1 \end{bmatrix} \tag{7}$$

Procedure (e) — The corrected odometry, p_i', is not represented in global coordinate. It has to be transformed to the global starting position, p_{om}, in order to obtain $\hat{p}$.

$$\begin{bmatrix} \hat{p} \\ 1 \end{bmatrix} = {}^{p_{om}}_{p_{oi}}T \begin{bmatrix} p_i' \\ 1 \end{bmatrix} \tag{8}$$

IV. Example Code of an Add-on Function for Sensor Fusion

The proposed algorithm is designed for the case that users do not want to change the existing control program of the mobile robot. In this section, an add-on function for sensor fusion is illustrated. The add-on function uses only odometry from the existing control program and global position from an equipped global position. The output of the add-on function is corrected global position. The code in figure 7 is written in MATLAB style.

All variables and return values of the functions are 3 x 1 column vectors. The function, trans(), represents the Eq. (4) as following,

$$\begin{bmatrix} result \\ 1 \end{bmatrix} = {}^{to}_{from}T \begin{bmatrix} this \\ 1 \end{bmatrix} \Leftrightarrow result = trans(from, to, this) \tag{9}$$

When the transform is represented as a function, the whole procedures can be expressed in one equation by recurrent function calls (See line 13.) For better performance, it is recommended to stabilize the p_{om} on the line 3 by filtering the measured global position before beginning sensor fusion.

```
1.  %store the initial variables at the time the sensor
    fusion starts
2.  poi = get_odomometry();
3.  pom = get_global_position();

4.  while sensor_fusion==ON,

5.  pi      = get_odomometry();
6.  pm      = get_global_position();

7.  %pm2    = trans(pom,poi,pm);
8.  %mu     = trans(pi,pm2,poi);
9.  %xi     = filter(mu);
10. %pi2    = trans(poi,xi,pi);
11. %p      = trans(poi,pom,pi2);

12. %substitute each expression for the variables
13. p       = trans(poi,pom,trans(poi,filter(trans(pi,
            trans(pom,poi,pm),Poi)),Pi));
14. end

15. function result = trans(from,to,this)

16. result(1,1) =
            to(1)+(this(1)-from(1))*cos(to(3)-from(3))-(
            this(2)-from(2))*sin(to(3)-from(3));
17. result(2,1) =
            TO(2)+(this(1)-from(1))*sin(to(3)-from(3))+(
            this(2)-from(2))*cos(to(3)-from(3));
18. result(3,1)=to(3)+this(3)-from(3);
```

Fig. 7. An example code for sensor fusion by using the proposed method. poi = p_{oi}, pom = p_{om}, pi = p_i, pm = p_m, pm2 = p_m', mu = μ, xi = ξ, pi2 = p_i', p = $\hat{p}$.

V. Control Using the Corrected Odometry

Using the corrected position, we can design any control to make the robot perform a desired behavior. According to the hardware setup, the input to the robot can be a desired position, desired velocity, or desired force. The desired velocity or desired can be used without regard to the proposed transform method because they are represented in the body frame of the robot. However, in the case that the control program of the purchased robot provides functions following the desired odometry. We have to transform the desired odometry to be compatible to the corrected odometry as shown in figure 8. Otherwise, the command can be applied to wrong direction.

When the desired global position, $\hat{p}_d$, is defined, it should be transformed to the odometry frame.

$$\begin{bmatrix} p_{id}' \\ 1 \end{bmatrix} = {}^{p_{oi}}_{p_{om}}T \begin{bmatrix} \hat{p}_d \\ 1 \end{bmatrix} \tag{10}$$

Because the robot control program believes that the robot is at the un-corrected odometry, p_i, the desired odometry, p_{id}', has to be transformed from the corrected odometry, p_i', to the un-corrected odometry, p_i.

$$\begin{bmatrix} p_{id} \\ 1 \end{bmatrix} = {}^{p_i}_{p'_i}T \begin{bmatrix} p'_{id} \\ 1 \end{bmatrix} \tag{11}$$

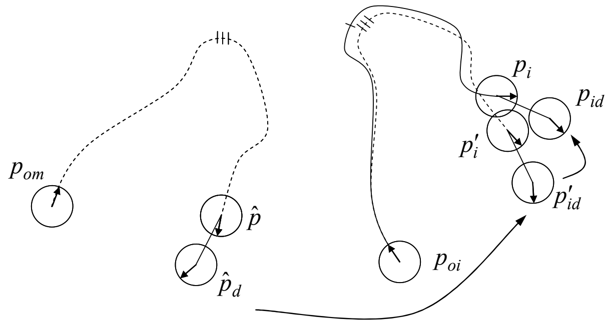

Fig. 8. The desired global trajectory should be transformed to odometry frame. This transform is required only for robots that demand desired odometry.

VI. Experiment

In order to test the algorithm, we use a mobile manipulator, Juliet, and a global localization program, CARMEN, which is outstanding SLAM software developed in Stanford AI Laboratory. In this experiment, only global position information was received from CARMEN. Juliet, the mobile manipulator, has a XR4000 omni-directional mobile base. The localization system gives the global position with a resolution of several centimeters and several degrees. The maximum fluctuation of the global position from the CARMEN is around 20cm. The control input is the desired odometry obtained by Eq. (11). The block diagram of the sensor fusion of this system is shown in figure 10. The function for getting odometry, XR4000_Getodometry(), consists of reading encoder of each wheel, calculating Jacobian for velocity, and integrating the velocity for odometry. Note that CARMEN uses the uncorrected odometry, p_i, not p'_i because it makes the global position with due regard to the errors of odometry by itself. The low-pass filter used here is a simple second order filter which is critical damped in order to make the filtered output not fluctuate. Second order low-pass filter can provide continuous velocity when the output is differentiated for PD control. The cut-off frequency is 1/5sec that is quite slow because the measured global position is very noisy.

$$\ddot{f}(x) + 2\zeta\omega\dot{f}(x) + \omega^2 f(x) = x$$
$$\text{where } \zeta: \text{ damping ratio} (=1) \tag{12}$$
$$\omega: \text{ cut-off frequency}$$

The target trajectory is a circle with a fixed angle. In order to confirm the efficiency of the proposed algorithm, an experiment without the use of corrected position information has also been performed. Figures 11 and 12 show the experimental result. The trajectory is as follows:

$$p'_{id}(t) = \begin{bmatrix} R\cos(\omega t) - 1 \\ R\sin(\omega t) \\ 0 \end{bmatrix} \tag{13}$$

where R, ω, and t are the radius of motion, angular velocity, and time, respectively.

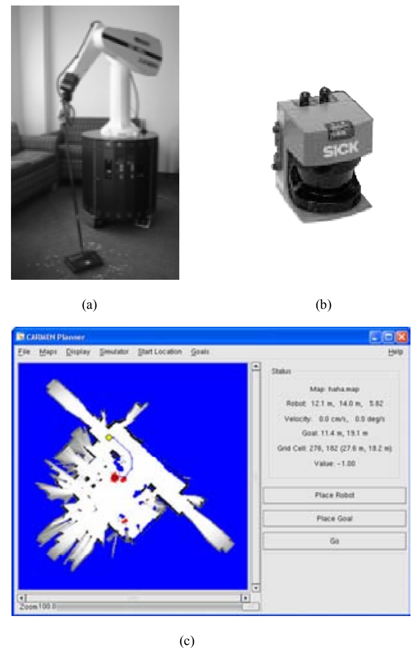

Fig. 9. (a) Mobile manipulator, Julet (b) Laser range sensor (c) CARMEN: global localization program of Stanford AI Laboratory

The experiment was carried out in two modes. When the correction mode is off, the odometry follows the trajectory of a perfect circle, and the global position starts to drift to top-right direction. Because the floor is carpeted, the amount of slippage is quite large. When the sensor fusion is activated, the robot moves along the corrected position information, $\hat{p}$, as described in figure 2. Although the measured global position is very noisy (solid line), it is possible to see that the robot follows the trajectory in the sensor fusion mode. The Corresponding odometry gradually drifts to compensate the drifting error (dashed line).

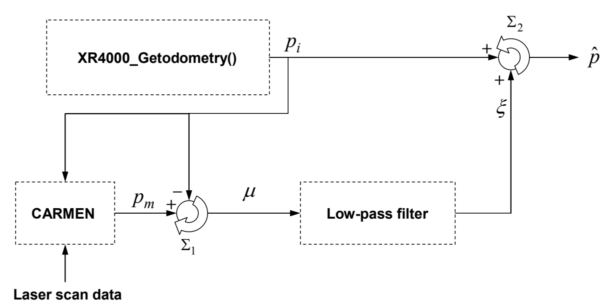

Fig. 10. Experimental setup. The functions to get the odometry and the global localization are not modified. Only feedforward low-pass filter and accompanying transforms are added for sensor fusion.

The maximum position errors in correction mode are

9.2cm in x direction, 10.7cm in y direction, and 6.2° in angle. The maximum position errors without the sensor fusion are 87.3cm in x direction, 36.1cm in y direction, and 19.8° in angle. The errors increase continuously with time. However the errors are determined using the measured global position which is very noisy. More accurate error analysis requires another accurate external sensor system. The movies of this experiment corresponding to Figs. 11 and 12, are available on the web:

http://robot.kaist.ac.kr/~leekb/aim2005movie.wmv
http://robot.kaist.ac.kr/~leekb/aim2005graph.wmv

VII. CONCLUSIONS

The proposed algorithm provides a convenient method to get more accurate global position and odometry by sensor fusion with global position sensor. The basic structure of this method uses feedforward type filter in order not to change the existing control program of mobile robot. This algorithm is suitable for tasks in which trajectory-following performance is critical because the corrected global position has good resolution at sub-milimeter level and experiences no drift. The sensor fusion mode can be activated or deactivated at any time due to the introduction of the starting positions, p_{om} and p_{oi}.

ACKNOWLEDGMENT

We thank Prof. Sebastian Thrun at Stanford AI laboratory for his assistance in using the SLAM software, CARMEN. We also thank Peter Thaulad for his assistance in implementing the hardware for the XR4000 robotic system.

REFERENCES

[1] P. Maybeck, “Stochastic models, estimation, and control,” New York: Academic Press, Vol. 1, 1979

[2] J. Borenstein and L. Feng, “Correction of Systematic Odometry Errors in Mobile Robots,” *Proceedings of the 1995 International Conference on Intelligent Robots and Systems (IROS '95),* August 5-9, pp. 569-574.

[3] J. Borenstein and L. Feng, "Measurement and Correction of Systematic Odometry Errors in Mobile Robots*." IEEE Journal of Robotics and Automation,* Vol 12, No 6, December 1996, pp. 869-880.

[4] J. Borenstein, L. Feng, and D. Wehe, “Mobile Robot Positioning Sensors and Techniques,*” Journal of Robotic Systems, Special Issue on Mobile Robots.* Vol. 14 No. 4, pp. 231 – 249.

[5] P. Maechler, "Robot Odometry Correction Using Grid Lines on the Floor", *Proceedings of MPCA,* Pisa, Italy, 1997

[6] A. Martinelli, N. Tomatis, A. Tapus, and R. Siegwart, "Simultaneous Localization and Odometry Calibration for Mobile Robot," Proceedings *of the 2003 IEEE/RSJ Intl. Conference on Intelligent Robots and Systems* Las Vegas, 2003

[7] A. Martinelli and R. Siegwart, "Estimating the Odometry Error of a Mobile Robot during Navigation," *Proceedings of European Conference on Mobile Robots,* 2003

[8] R. Kelley, I. Katz, and C. Bedoya, “Design, development and evaluation of an Ada coded INS/GPS open loop Kalman filter,” Aerospace and Electronics Conference, 1990. NAECON 1990., Proceedings of the IEEE 1990 National , vol.1, Pages:382 - 388,1990

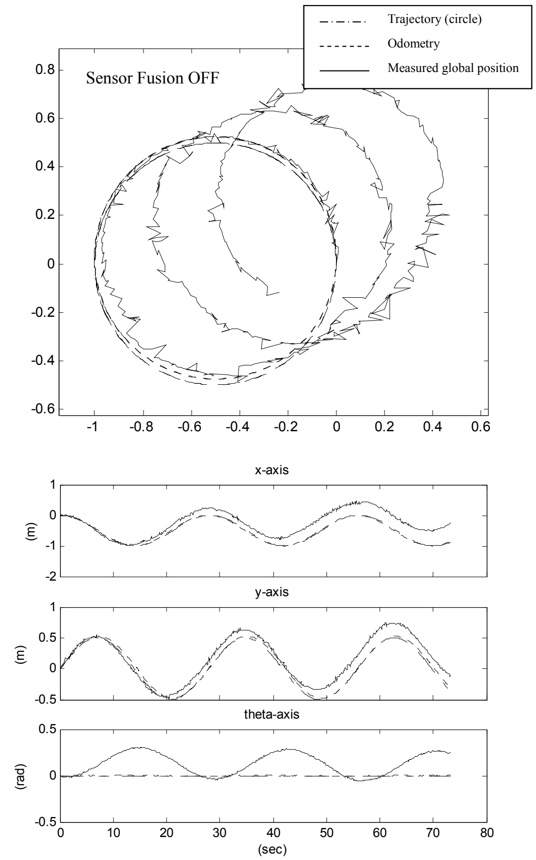

Fig. 11. Experimental result: without sensor fusion

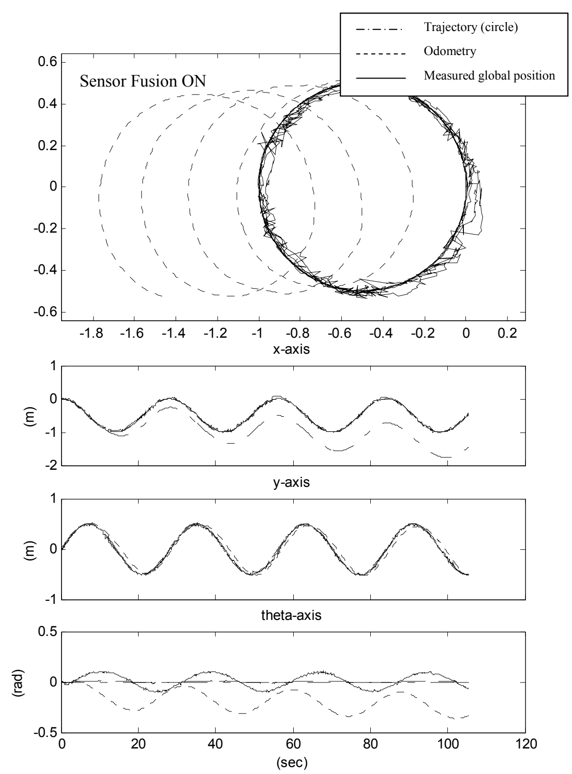

Fig. 12. Experimental result: with sensor fusion.