2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

Remote control of a moving robot using the virtual link

Young Soo Suh, Sang Kyeong Park, Dae-Nyeon Kim, and Kang-Hyun Jo

***Abstract*— A new remote control method of a moving robot is proposed, where a moving robot is moved according to the pointing position and orientation of the remote controller. The remote controller consists of the camera and gyroscopes. The landmark in a moving robot is recognized by the camera in the remote controller and a robot is moved in the camera's pointing position and orientation. The 'virtual link' term is used since the robot is moved as if there is a link between the robot and the remote controller. Gyroscopes are also used in the remote controller so that fast estimation of the camera's pointing position and orientation is possible. The proposed method is verified through experiments.**

I. INTRODUCTION

Most popular remote controller for wheeled mobile robots is the joy stick type, which is commonly used for playing games or RC products. While joy stick type remote controllers are easy to learn and use, it is sometimes difficult to control wheeled mobile robots using them. For example, if the path for the mobile robot is complicated, it is not easy to remotely control it using the joy stick type remote controller. Another difficult situation is to make the mobile robot follow or lead a moving human, who has the remote controller.

For these reasons, different types of remote controllers have been proposed. In [1], the Exosckeleton type motion capturing device is used as the remote controller. A wheeled robot is intuitively controlled based on the movement of human arms, which is detected by the Exosckeleton type motion capturing device. In [2], automatic tracking of a human being by a mobile robot is proposed, where an omnidirectional camera and a laser sensor is installed in a mobile robot. Also for complicated path following, haptic-feedback [3] is used for more easy remote control.

In this paper, we propose a new type of remote controllers, where a camera and gyroscopes are used in the remote controller. A landmark is placed in the mobile robot. From the landmark vision, a camera center point projection (both position and orientation) is computed and the robot is moved according to the computed position and orientation. Thus if a person moves and rotates the remote controller, the moving robot is moved as if there is a bar or a link connected between the remote controller and the moving robot. The term *virtual link* is used to emphasize this feature. A camera alone cannot maintain the virtual link when the remote controller is rotating fast; to handle this problem, gyroscopes outputs are combined with camera vision.

Key technologies are estimation of position and orientation of the remote controller and control algorithm of the mobile robot. L. Naimark and E. Foxlin [4] provide a fairly complete solution to the position and orientation estimation problem using the landmark in the context of motion tracking. In this paper, we use simpler landmark for ease and robustness of landmark tracking. In [5], control algorithm for the mobile robot is discussed when the target is given. The control objective is slightly different in this paper since the destination is constantly changing as the remote controller is moving.

The paper is organized as follows. In Section II, the system overview is given. In Section III, how to estimate the remote controller's pointing position and orientation is described. The estimated position and orientation become target position and orientation of the mobile robot. In Section IV, how to move the mobile robot to the target position and orientation is discussed. To verify the proposed method, experiments are done in Section V and conclusion is given in Section VI.

II. SYSTEM OVERVIEW

System overview is given in Fig. 1. The remote controller consists of the camera and 3 gyroscopes. If the camera sees four circles in the mobile robot, the camera position and orientation with respect to the robot coordinate can be computed [6].

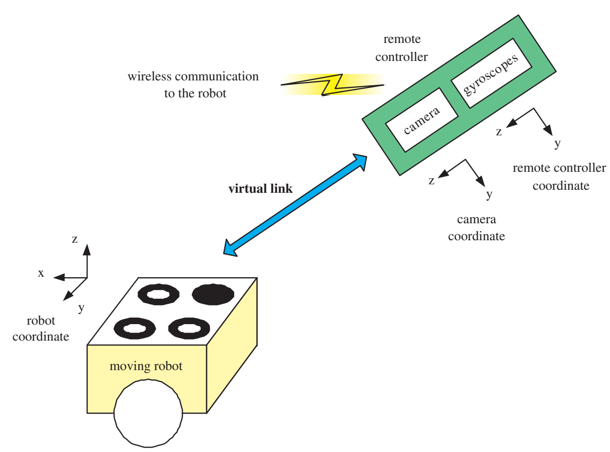


Fig. 1. Remote Control System Overview

From the camera position and orientation information, we can compute projection of $-y$ axis (the camera coordinate)

This work was supported by grant No. R01-2006-000-11334-0 from the Basic Research Program of the Korea Science Engineering Foundation and Network-based Automation Research Center (NARC) at University of Ulsan. The authors would like to express financial supports from post-BK(Brain Korea) 21 program.

Dept. of Electrical Engineering, University of Ulsan, Namgu, Ulsan, Korea. suh@ieee.org

direction vector to the $z = 0$ plane (the robot coordinate): see Fig. 2. Note that the most natural direction of the robot movement is in the projected direction of $-y$ axis (camera coordinate). The origin of this projection vector is denoted by (x_t, y_t) (in the robot coordinate): this point is where the camera origin is pointing. The angle ψ_t is defined as the angle between the projected vector and the x axis (the robot coordinate).

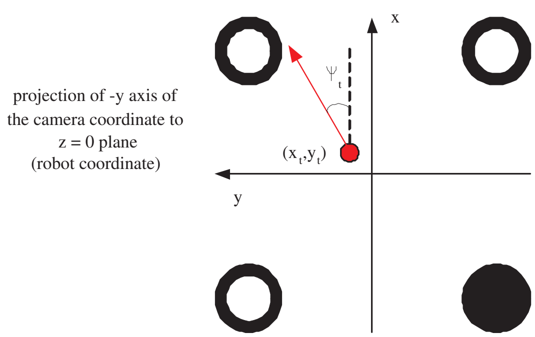


Fig. 2. Projection of $-y$ camera axis to the mobile robot

Once x_t, y_t, ψ_t are computed and transmitted to the robot, the robot moves so that the robot's position and orientation coincide with (x_t, y_t) and ψ_t. This control strategy enables a person to remotely control a mobile robot intuitively; just point the remote controller to the direction and location you want to move, then the robot will move to the direction and location.

III. Remote Controller's Pointing Position and Orientation Estimation

When the perspective projection is used, at least four points are necessary for position and orientation estimation of the camera [6]. We use circle type landmark as in Fig. 1. The following is a brief description of the sequence of operations used to identify four points (center of circles).

- Noise reduction : Gaussian filter is applied to the 640×480 image to reduce the noise
- Edge detection : Canny filter is used for edge detection.
- Circle finding : Only proper size circle features are selected testing the size (edge boundary rectangle size smaller than 5 pixels or larger than 150 pixels are thrown out) and the area ratio of the edge and the boundary rectangle is tested (the ratio should be close to $\pi/4$).
- Coaxial circle finding : Two circles with a white inner circle and a black outer circle, whose centers are almost the same, are selected. There should be three two-coaxial-circles.
- Remaining one black circle finding : Using three two-coaxial-circles, windows of remaining one black circle candidate location are computed and search the proper size (similar to the outer coaxial circle) black circle within the windows.
- Four circle identification : If four circles with three two-coaxial-circles and one black circle are found, centers of four circles are computed. The black circle is labeled as circle 1 and the others are labeled as circle 2, 3, 4 counter-clock-wise, respectively.

Let $P_{k,r}$ $(k = 1, 2, 3, 4)$ be the centers of four circles in the robot coordinate and $P_{k,i}$ and $P_{k,c}$ be the corresponding points in the image coordinate and the camera coordinate, respectively:

$$P_{k,r} = \begin{bmatrix} X_k \\ Y_k \\ 0 \end{bmatrix}_r, \quad P_{k,i} = \begin{bmatrix} x_k \\ y_k \end{bmatrix}_i, \quad P_{k,c} = \begin{bmatrix} s_k x_k \\ s_k y_k \\ s_k \end{bmatrix}_c.$$

Once the four points landmark is identified, the camera position and orientation can be computed : i.e., we can compute the rotation matrix $R \in \mathcal{R}^{3\times3}$ and the transition vector $t \in \mathcal{R}^{3\times1}$, which satisfy the following relationship:

$$\begin{bmatrix} X_k \\ Y_k \\ 0 \end{bmatrix}_r = R \begin{bmatrix} s_k x_k \\ s_k y_k \\ s_k \end{bmatrix}_c + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}_r. \quad (1)$$

Once R and t are computed, (x_t, y_t) (the point in the $z = 0$ robot coordinate plane, where the origin of the camera is pointing) can be computed as follows

$$\begin{aligned} x_t &\triangleq t_x - t_z \tfrac{R_{1,3}}{R_{3,3}} \\ y_t &\triangleq t_y - t_z \tfrac{R_{2,3}}{R_{3,3}}. \end{aligned} \quad (2)$$

The relationship (2) is from the following:

$$\begin{bmatrix} x_t \\ y_t \\ 0 \end{bmatrix}_w = R \begin{bmatrix} 0 \\ 0 \\ s_t \end{bmatrix}_c + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}_r.$$

To compute ψ_t, we compute (x_p, y_p) from the following:

$$\begin{bmatrix} x_p \\ y_p \\ 0 \end{bmatrix}_w = R \begin{bmatrix} 0 \\ -1 \\ s_p \end{bmatrix}_c + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}_r. \quad (3)$$

Note that $[\ x_p - x_t,\ y_p - y_t,\ 0\ ]'_r$ is the projected direction of the camera $-y$ axis to the $z = 0$ robot coordinate plane. From (2) and (3), we obtain

$$\begin{aligned} x_p - x_t &= -R_{1,2} + \tfrac{R_{3,2}}{R_{3,3}} R_{1,3} \\ y_p - y_t &= -R_{2,2} + \tfrac{R_{3,2}}{R_{3,3}} R_{2,3}. \end{aligned} \quad (4)$$

Now ψ_t can be computed as follows:

$$\psi_t \triangleq \operatorname{atan2}(y_p - y_t, x_p - x_t). \quad (5)$$

The camera position and orientation from vision data can be computed as fast as 10 Hz sampling rate. Actual sampling rate could be lower than 10 Hz if the remote controller or the mobile robot moves fast so that the landmark recognition success rate is falling. For more rigid virtual link, three gyroscopes are used, whose function is to provide higher sampling rate estimation of the orientation. The high sampling rate estimation of position using double integration of accelerometer outputs is also possible. However, it is a more difficult task and remains as a future work. We note that small change in the camera position only results in small movement of a mobile robot. On the other hand, small change in the camera orientation could result in large movement of a mobile robot. This can be seen in (2), where the small change in $R_{3,3}$ could result in large change in x_t and y_t. Thus fast estimation of the camera orientation is more important than that of the camera position estimation.

The Euler angles are used as the camera orientation parameters (with respect to the mobile robot coordinate):

$$a \triangleq \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} \tag{6}$$

where the Euler rotation sequence is as follows: ψ (z axis) $\rightarrow \theta$ (y axis) $\rightarrow \phi$ (x axis) sequences.

Note that the Euler angles and the rotation matrix R are related as follows [7] :

$$R = \begin{bmatrix} \cos\psi\cos\theta & \sin\psi\cos\theta & -\sin\theta \\ \cos\psi\sin\theta\sin\phi - \sin\psi\cos\phi & \sin\psi\sin\theta\sin\phi + \cos\psi\cos\phi & \cos\theta\sin\phi \\ \cos\psi\sin\theta\cos\phi + \sin\psi\sin\phi & \sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi & \cos\theta\cos\phi \end{bmatrix}. \tag{7}$$

In the Kalman filter, we need to compute the Euler angles from the rotation matrix R (derived from the vision data). When R is obtained from the vision data, it is satisfied $-\pi/2 < \theta < \pi/2$, which means the camera is pointing downward. If the camera is pointing upward, R cannot be obtained from the vision data. Computing the Euler angles from (7) is done as follows:

$$\begin{aligned} \theta &= \sin(-R_{1,3}) \\ \phi &= \operatorname{atan2}(R_{3,3}, R_{2,3}) \\ \psi &= \operatorname{atan2}(R_{1,1}, R_{1,2}). \end{aligned}$$

Let $\omega_c \in \mathcal{R}_{3\times1}$ be the angular velocity with respect to the camera coordinate and $\omega_g \in \mathcal{R}_{3\times1}$ be the gyroscope outputs:

$$\omega_g = R_{r\rightarrow c}\omega_c + v_g \tag{8}$$

where $R_{r\rightarrow c}$ is a constant rotation matrix between the camera coordinate and the remote controller coordinate.

Then the following relationship is satisfied [8]

$$\dot{a} = \mathrm{Eul}(a)\omega_c \tag{9}$$

where

$$\mathrm{Eul}(a) \triangleq \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix}.$$

From (8) and (9), we have

$$\dot{a} = \mathrm{Eul}(a)R_{r\rightarrow c}(\omega_g - v_g) \tag{10}$$

By integrating both sides using the following approximation

$$\begin{aligned} &\textstyle\int_{kT}^{(k+1)T} \mathrm{Eul}(a)R_{r\rightarrow c}v_g\,dt \approx \\ &\quad T\,\mathrm{Eul}(a(kT)))R_{r\rightarrow c}v_g(kT) \\ &\textstyle\int_{kT}^{(k+1)T} \mathrm{Eul}(a)R_{r\rightarrow c}\omega_g\,dt \approx \\ &\quad T\,\mathrm{Eul}(a(kT))R_{r\rightarrow c}\omega_g(kT) \end{aligned}$$

we have the following discretized equation:

$$a[k+1] = a[k] + \tilde{v}[k] + T\,\mathrm{Eul}(a[k])R_{r\rightarrow c}\omega_g[k] \tag{11}$$

where

$$\begin{aligned} &a[k] \triangleq a(kT) \\ &\mathrm{E}\{\tilde{v}[k]\tilde{v}[k]'\} = T^2\,\mathrm{Eul}(a[k])R_{r\rightarrow c}\mathrm{E}\{v_g v_g'\}R_{r\rightarrow c}'\,\mathrm{Eul}(a[k])'. \end{aligned}$$

The sampling period of gyroscope output is $T \triangleq 10ms$ and the sampling period of the vision data is $NT = 100ms$ with $N \triangleq 10$. Due to the image processing time, the $a(kT)$ estimation from the vision is available at time $(k+M)T$ with $M \triangleq 5$.

The measurement equation is given as follows:

$$a_{vision}[k] = a[k-M] + v_{vision}[k-M] \tag{12}$$

where a_{vision} is the Euler angles computed from the vision data and v_{vision} is the measurement noise of the vision. We assume that v_{vision} is zero-mean white Gaussian noise and $\mathrm{E}\{v_{vision}v_{vision}'\} = R$. We also assume that v_{vision} and ω_g are uncorrelated.

Note that $a_{vision}[k]$ is available at most every $100ms$. When $a_{vision}[k]$ is not available, we set $R = \infty$.

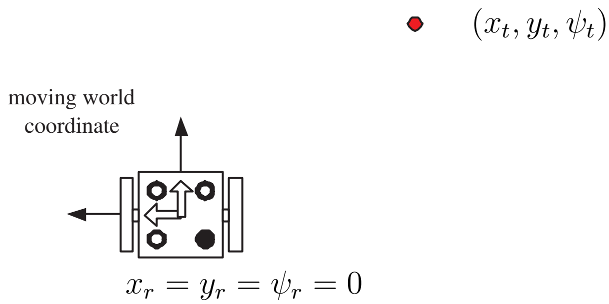
We applied the Kalman filter to model (11) and (12) to obtain the camera's position and orientation estimation. From this estimation, we can compute (x_t, y_t) and ψ_t using (2) and (5). We note that $R_{i,j}$ can be computed from (7).

The computed (x_t, y_t) and ψ_t are transmitted to the mobile robot through wireless communication. As a safety measure, the stop command is transmitted if $|\hat{\theta}| > \pi/2$, which means the remote controller is pointing upward. Thus the robot can be stopped anytime by just pointing the remote controller upward.

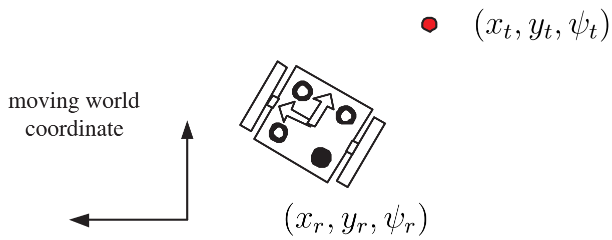
IV. Mobile Robot Control Algorithm

Control objective is to move the mobile robot so that its position and orientation track the target position (x_t, y_t) and orientation ψ_t, which is pointing position and direction of the remote controller. Instead of using the fixed world coordinate, we use moving world coordinates. When the vision data is available, the moving world coordinate is defined as the robot coordinate at that time. This moving world coordinate does not change until the next vision data is available. The robot position and orientation is defined with respect to this moving world coordinate: the position is denoted by (x_r, y_r) and ψ_r. We note that (x_r, y_r) and ψ_r are computed from the dead reckoning of the robot. The robot position and orientation is set to $x_r = y_r = \psi_r = 0$ when new vision data is available and thus new moving world coordinate is assigned.

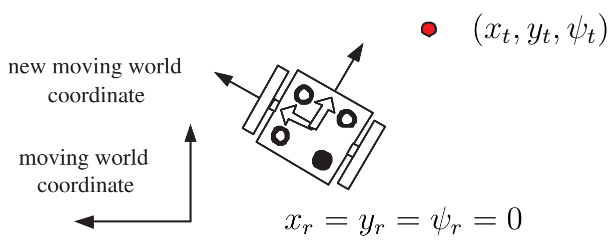
In Fig. 3, the concept of the moving world coordinate is illustrated. In (a), the moving world coordinate is assigned since the new vision data is available. In (b), the robot is moving toward the target position and orientation. In this period, (x_r, y_r) and ψ_r is changing and the moving world coordinate does not change. In (c), another new vision data is available and thus new moving world coordinate is assigned as the robot coordinate at the time. The robot position is reset so that $x_r = y_r = \psi_r = 0$ since the robot is in origin with respect to the new moving world coordinate.



(a) the robot coordinate at the time vision data is available becomes the (moving) world coordinate



(b) (x_r, y_r, ψ_r) is changing as the robot moves



(c) new vision data is available and the new (moving) world coordinate is defined

Fig. 3. World coordinate is changing whenever new vision data is available

There are a number of control algorithms (for example, [5], [9]) for mobile robot posture stabilization, where both position and orientation of robot is controlled. In our application, the target position and orientation is constantly changing as the user is moving the remote controller. In particular, we don't need sophisticated path generation algorithms since the target is usually near and the target is likely to move away to somewhere else before the robot is reaching the target. Hence we use a simple control algorithm.

Let u_l and u_r be the control input to the left and right motor. The kinematic model of a two wheeled robot is

$$\begin{aligned} \dot{x}_r &= k_1 \cos\psi_r u_p \\ \dot{y}_r &= k_1 \sin\psi_r u_p \\ dotpsi_r &= k_2 u_o \end{aligned} \tag{13}$$

where

$$u_p \triangleq \frac{u_r + u_l}{2}, \quad u_o \triangleq \frac{u_r - u_l}{D}.$$

The parameter D is the length between two wheels and k_1 and k_2 are constants related to motors. The motor dynamics is ignored since they are much faster than mobile robot dynamics: thus, the right (left) motor speed is assumed to be proportional to u_r (u_l). We use u_p (which control the position) and u_o (which control the orientation) as control inputs.

Let the distance to target d_{t-r} and the angle to the target ψ_{t-r} be defined by

$$\begin{aligned} d_{t-r} &\triangleq \sqrt{(x_t - x_r)^2 + (y_t - y_r)^2} \\ \psi_{t-r} &\triangleq \text{atan2}(y_t - y_r, x_t - x_r) \end{aligned}$$

The pseudo-code of the control algorithm is given as follows:

`if` $t_d > d_1$
    `if (` $|\psi_{t-r} - \psi_r| \leq \pi/2$ `)`
        `direction = 1`
        $u_o = k_{o1} \times (\psi_{t-r} - \psi_r + \pi)$
    `else`
        `direction = -1`
        $u_o = k_{o1} \times (\psi_{t-r} - \psi_r)$
    `end`
    $u_p = direction \times k_p \times d_{t-r}$
`else`
    $u_p = 0$
    $u_o = k_{o2} \times (\psi_t - \psi_r)$
`end`

In the above algorithm, the angle computation result is always adjusted so that the result is in the region of $(-\pi, \pi]$. For example, if $\psi_{t-r} = 0.8\pi$ and $\psi_r = -0.5\pi$, then $\psi_{t-r} - \psi_r = -0.7\pi$.

The proposed control strategy is simple. Until the robot is near the target (*i.e.*, the distance to target is less than d_1), the robot moves to the target direction. If the robot is near the target, the robot rotates until $\psi_t = \psi_r$.

V. Experiments

The remote controller is given in Fig. 4: IEEE 1394 camera (Point Grey's 640 x 480 dragonfly) and three orthogonally aligned ADXRS150 (Analog Device) gyroscopes are used.

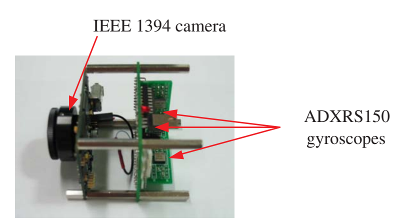


Fig. 4. Remote Controller Structure

Two wheel mobile robot is used as in Fig. 5. The forward direction is x axis direction of the robot coordinate.

The first experiment is to test the remote controller's orientation estimation. In Fig. 6, orientation estimation from 3 different experiments are given. In the first graph, we have mainly changed ϕ while θ and ψ movement is suppressed.

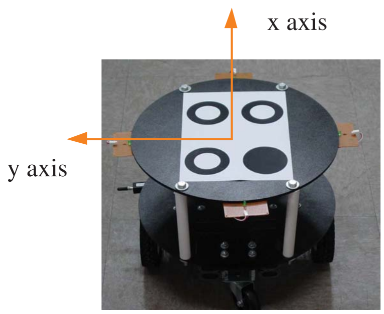


Fig. 5. Mobile Robot

The x mark in the graph is estimated using the vision and the solid line is orientation estimate from the Kalman filter. It can be seen that the Kalman filter estimate almost coincides with the vision data when it is available, which means that the Kalman filter output accurately estimate the orientation. In the second graph, θ is mainly changed and in the third graph ψ is mainly changed.

The second experiment is to test whether the mobile robot is able to track the remote controller's movement. The remote controller is attached at the encoder so that it can be rotated with hands (see Fig. 7). In this experiment, a one-dimensional coordinate system is introduced, where the origin is where the remote controller is. The remote controller's pointing position is denoted by X_t and the robot's position is denoted by X_r. Note that X_t and X_r are different from x_t and x_r: x_t is defined with respect to the robot coordinate and x_r is defined with respect to the moving world coordinate.

The remote controller is moved with hands so that the remote controller's pointing position (X_t) increases. In Fig. 8, the upper graph is X_t and X_r plot, which are computed from the vision data. When the vision data is not available, the previously computed values are used. Thus the constant value means that the vision data is not available. This is usually because the remote controller cannot recognize the landmark due to the fast movement of the mobile robot. The lower graph is $X_t - X_r$ plot. If $X_t - X_r < d_1 = 70(mm)$, the mobile robot is not moved to prevent too sensitive motion. The remote controller began to move at time 3.2 sec and it can be seen that the robot (X_r) is tracking the remote controller's pointing position (X_t)

To see the roles of gyroscopes, time interval between 3.2 sec and 3.9 sec is magnified in Fig. 9. The camera captures images every 100 ms but it could not recognize the landmark at 3.3, 3.5, 3.6, 3.7, 3.8 seconds since at this time the remote controller's orientation is changing fast and the mobile robot is moving fast. As can be seen in Fig. 9, during this period the target position is computed in the Kalman filter using (11). Thus smooth remote control is possible even if the vision data is not available.

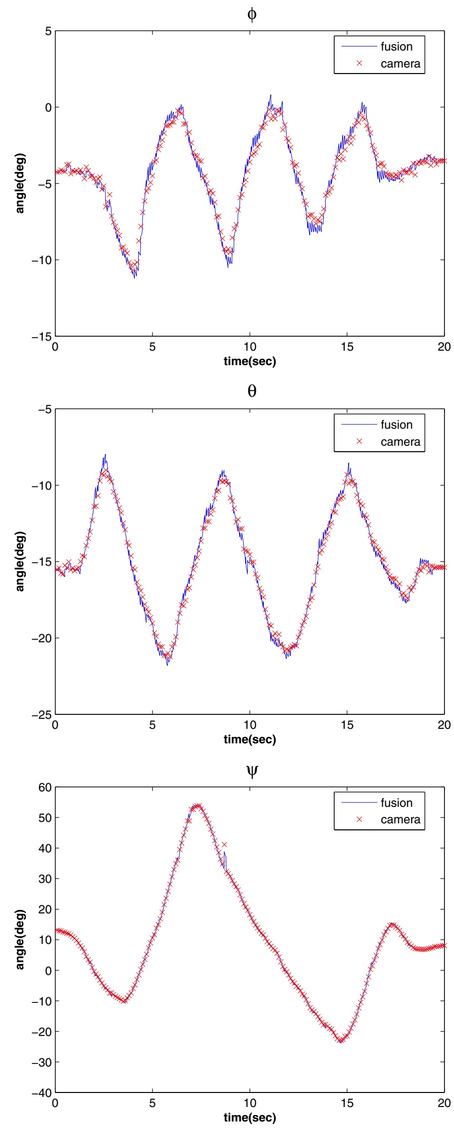


Fig. 6. Remote Controller's Orientation Estimation Test

VI. Conclusion

A new remote control method for mobile robots using the vision and inertial sensors (gyroscopes) are proposed. The key technology is combination of vision and inertial data so that the mobile is moved by simply pointing the remote controller. Experiments have been performed in order to validate the proposed method. This method could be applied to semi-automatic control of cleaning robots and automatic leading/following of a robot to the human operator. Future research topic is to include accelerometer output, which could enable more robust and smooth remote control of mobile robots.

References

[1] P. W. Jeon and S. Jung, "Teleoperated control of mobile robot using exoskeleton type motion capturing device through wireless communication," in *Proc. of 2003 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pp. 1107–1112, 2003.

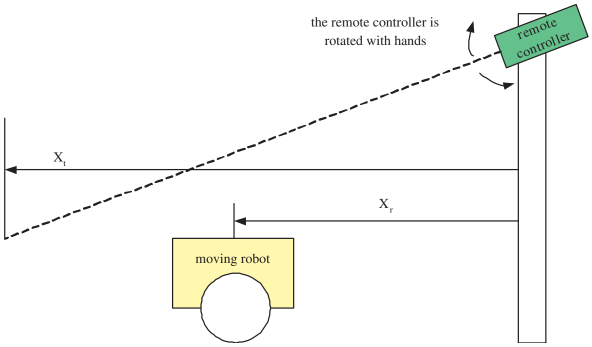


Fig. 7. Experiment Setting for One-directional Tracking

[2] M. Kobilarov, G. Sukhatme, J. Hyams, and P. Batavia, "People tracking and following with mobile robot using an omnidirectional camera and a laser," in *Proc. of the 2006 IEEE International Conference on Robotics and Automation*, (Orlando, Florida), pp. 557–562, 2006.

[3] D. Lee, O. Martinez-Palafox, and M. W. Spong, "Bilateral teleoperation of a wheeled mobile robot over delayed communication network," in *Proc. of the American Control Control Conference*, (Denver, U.S.A.), pp. 2887–2892, 2003.

[4] L. Naimark and E. Foxlin, "Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker," in *IEEE International Symposium on Mixed and Augmented Reality (ISMAR 2002)*, pp. 27–36, 2002.

[5] S. Lee, Y. Youm, and W. Chung, "Control of car-like mobile robots for posture stabilization," in *Proc. of the 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1745–1750, 1999.

[6] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[7] J. B. Kuipers, *Quaternions and rotation sequences: a primer with applications to orbits, aerospace, and virtual reality*. New Jersey: Princeton University Press, 1999.

[8] R. C. Nelson, *Flight Stability and Automatic Control*. New York: McGraw-Hill, 2nd edition ed., 1998.

[9] R. M. Murray and S. S. Sastry, "Steering nonholonomic systems in chained form," in *Proc. of the 30th IEEE Conference on Decision and Control*, pp. 1121–1126, 1991.

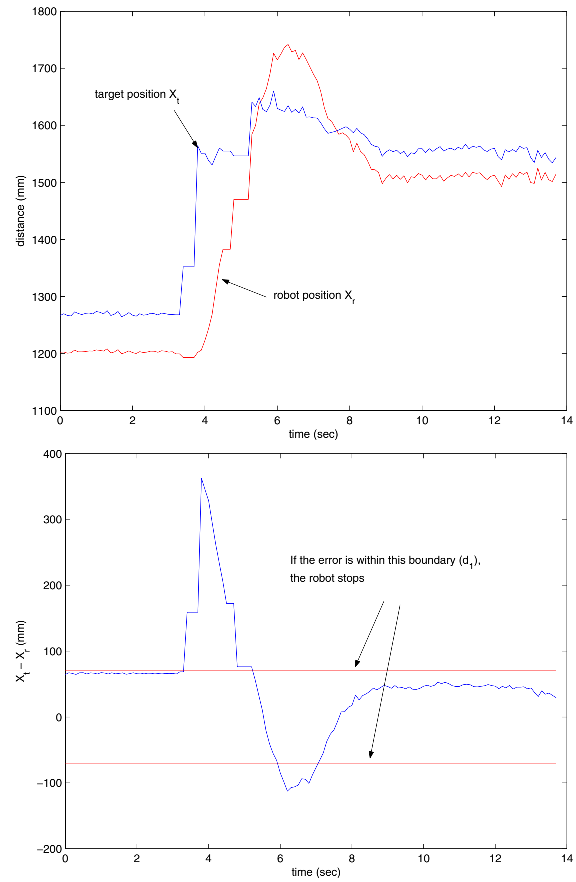


Fig. 8. Tracking of the target position

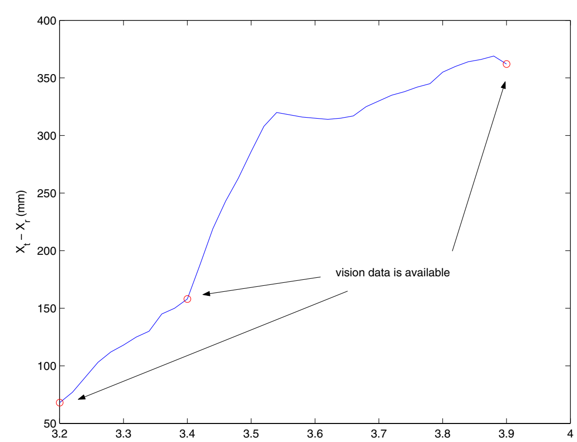


Fig. 9. When vision data is not available, gyroscopes estimate the target position