Proc. 2009 Int. Symp. on Robot and Human Interactive Communication,
pp. 1198-1203, Toyama, Japan, Sep./Oct. 2009.

Development of a Vision-Based Interface for Instructing Robot Motion

Junichi Sugiyama and Jun Miura

***Abstract*— This paper describes a vision-based interface for instructing robot motion easily. An interface that makes the robot move in the same way as user's motion is effective for an intuitive motion instruction. Such an interface can be realized by estimating the pose (position and orientation) of the interface and executing move commands for making the robot take the same pose. We estimate the pose of the interface by a monocular SLAM method, which is based on visual features and the extended Kalman filter. By additionally using an orientation sensor and an accelerometer, the reliability and the accuracy of pose estimation are improved. From the estimated pose, the target values of the robot position and the head orientation are set and the robot moves to achieve them. We implemented an experimental system which run in real-time (30 [Hz]) and successfully applied it to controlling a humanoid robot.**

I. INTRODUCTION

Service robots are expected to help human in various everyday situations. Possible tasks of such robots are: bringing a user-specified object, cleaning a room, mobile aid, social interaction. One of the important issues in using such a robot is teaching, because service robots need to work in various environment and, therefore, it is difficult to give a robot a complete set of required skills and knowledge in advance. This paper deals with instruction of robot motion.

Several methods for robot teaching have been developed which measure a human demonstration and convert it to robot commands [1], [2]. These works mainly focus on instructing robot's arm and hand motions. In this paper, we take such an indirect approach but deal with on-line instructing a humanoid robot to follow a movement.

To instruct a robot motion indirectly, it is necessary to measure a human motion in some way and to give it to the robot as the target motion. In this paper, we develop a vision-based interface by which the user can instruct a robot motion intuitively by making the robot move in the same way as the user's motion; if the user moves forward (backward), the robot moves forward (backward) and if the user turns right (left), the robot turns its head to right (left). For this purpose, we wear an interface device and measure its motion using a camera and sensors embedded in the interface.

Several wearable systems for pose (position and orientation) estimation have been developed. Kourogi and Kurata [3] developed a system which estimates the pose by integrating data from several sensors attached to a body using Kalman filter. Known visual patterns placed in the environment are additionally used for pose correction. Harada *et al.* [4] developed a system for recognizing human motion. They attach sensor modules to the lower limbs to measure their motion thereby recognizing the motion type such as walking and jumping. Several sensors including gyroscope and accelerometers are used but visual information is not used. Maeda *et al.* [5] developed a sensor system called Behavioral Interface. It can measure full body motion of the user by wearing a lot of sensors.

This research uses vision as a primary sensor. We adopt MonoSLAM method developed by Davison *et al.* [6] which performs real-time SLAM using a monocular camera. Since vision information suffers from instability to rapid camera motions, we additionally use an orientation sensor and an accelerometer and integrate their data with vision data using extended Kalman filter (EKF). Data integration basically follows the formulation by Armesto *et al.* [7]. The estimated human pose is used to issue a command for moving a humanoid robot. We implemented an experimental system which run in real-time (30 [Hz]) and successfully applied it to controlling a humanoid robot.

The rest of the paper is organized as follows. Sec. II describes an overview of the system. Sec. III explains the pose estimation algorithm using EKF. Sec. IV describes experimental results on SLAM and robot motion control. Sec. V concludes the paper and discusses future work.

II. SYSTEM OEVRVIEW

Fig. 1 is the interface which we developed. The interface is composed of a camera, a 3-axis orientation sensor, and a 3-axis accelerometer. It performs SLAM by integrating information from these sensors using EKF and commands a robot to move using the estimated pose.

The camera is Qcam Ultra Vision by Logicool and the 3-axis orientation sensor and accelerometer are embedded in InertiaCube3 by InterSense. These sensors are used for coping with a rapid motion of the interface which causes image blur.

Fig. 2 is the humanoid robot, Enon by Fujitsu, which has a wheeled platform and a rotatable head. The connection between the interface and the robot uses wireless LAN.

III. POSE ESTIMATION USING EKF

We basically use MonoSLAM developed by Davison *et al.* [6] for the pose estimation of the interface. This SLAM method uses only a monocular camera and estimates the pose of the camera and the 3D map of features around the camera. The state vector to estimate includes the pose of the

J. Sugiyama is with the Department of Information and Computer Sciences, Toyohashi University of Technology, 1-1 Hibarigaoka, Tenpaku, Toyohashi, Aichi, 441-8580, Japan. sugiyama@aisl.ics.tut.ac.jp

J. Miura is with the Department of Information and Computer Sciences, Toyohashi University of Technology, 1-1 Hibarigaoka, Tenpaku, Toyohashi, Aichi, 441-8580, Japan. jun@ics.tut.ac.jp

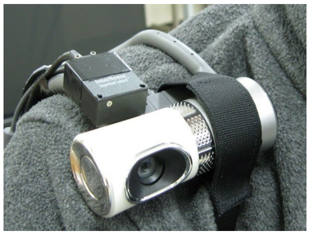


Fig. 1. Interface which has a camera, an orientation sensor, and an accelerometer.

Fig. 2. Humanoid robot, Enon by Fujitsu.

camera, the translational and rotational velocities, and the 3D positions of features in the map. The method requires some prior known features for defining the scale.

A feature is an image patch centered at a feature point such as an object corner detected by the interest operator of Shi and Tomasi [8]. The depth of a feature cannot be determined from a single image, hence the feature depth is initialized using parallax obtained by using a few subsequent frames. This initialization step is performed using 100 particles representing depth hypotheses along the 3D line from the position of the camera to the feature at the time of first detection. The particles which form uniform distribution as prior distribution exist in range $0.5\,[\mathrm{m}]$ to $5.0\,[\mathrm{m}]$ along the line. The initialization step finishes when the particle distribution converges enough so that it can be approximated by a 1D Gaussian.

Once a feature is detected in a camera image, the feature is matched in subsequent frames using the normalized SSD (sum of the squared differences) correlation to identify the estimated feature in the 3D map with the feature point in the 2D camera image. This matching is performed in the so-called 3σ region defined by the innovation covariance matrix in EKF, and therefore the matching fails if the camera move too fast.

To cope with this problem, we additionally use an orientation sensor and an accelerometer. Armesto *et al.* [7] developed a method of ego-motion estimation by integrating data from a camera and an inertial sensor using EKF with known image patterns (i.e., no map making). The state vector to estimate includes the pose of robot, the translational and rotational velocities, the translational acceleration, and the bias of the accelerometer. The motion model of the robot considers the tangential and centripetal accelerations. In addition, since the sampling rates for the sensors are different from each other, the method can estimate the robot pose using only vision, only an inertial sensor, or both sensors.

We combine the above two methods for a reliable pose estimation in a complex environment. That is, we use the motion model of [7] in the MonoSLAM framework [6].

A. Motion Model [7]

The state vector $\hat{\boldsymbol{x}}$ and its covariance matrix $\boldsymbol{P}$ to estimate is:

$$\hat{\boldsymbol{x}} = \begin{pmatrix} \hat{\boldsymbol{x}}_I \\ \hat{\boldsymbol{y}}_1^W \\ \hat{\boldsymbol{y}}_2^W \\ \vdots \end{pmatrix}, \quad \boldsymbol{P} = \begin{bmatrix} \boldsymbol{P}_{x_I x_I} & \boldsymbol{P}_{x_I y_1} & \boldsymbol{P}_{x_I y_2} & \cdots \\ \boldsymbol{P}_{y_1 x_I} & \boldsymbol{P}_{y_1 y_1} & \boldsymbol{P}_{y_1 y_2} & \cdots \\ \boldsymbol{P}_{y_2 x_I} & \boldsymbol{P}_{y_2 y_1} & \boldsymbol{P}_{y_2 y_2} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix},$$

where $\hat{\boldsymbol{x}}_I$ is the estimated state vector of the interface and $\hat{\boldsymbol{y}}_i^W$ $(i = 1, 2, \cdots)$ is the estimated position of a feature. The state vector of the interface $\boldsymbol{x}_I$ following [7] is:

$$\boldsymbol{x}_I = \begin{pmatrix} \boldsymbol{p}^{W\,T} & \boldsymbol{q}^{WI\,T} & \boldsymbol{v}^{W\,T} & \boldsymbol{\omega}^{W\,T} & \boldsymbol{a}^{W\,T} & \boldsymbol{b}^{W\,T} \end{pmatrix}^T,$$

where $\boldsymbol{p}^W$, $\boldsymbol{v}^W$, and $\boldsymbol{a}^W$ are respectively the position, the velocity, and the acceleration of translation of the interface in the world coordinate frame W, and $\boldsymbol{q}^{WI}$ and $\boldsymbol{\omega}^W$ are respectively the orientation and the angular velocity of the interface. $\boldsymbol{q}^{WI}$ is represented by quaternion and $\boldsymbol{\omega}^W$ declares that its norm is a rotation angle and its orientation is a rotation axis. $\boldsymbol{b}^W$ is the bias of the accelerometer.

The motion model of the interface is:

$$\boldsymbol{x}_I(k{+}1) = \begin{pmatrix} \boldsymbol{p}^W(k) + \boldsymbol{v}^W(k)\Delta t + \frac{\Delta t^2}{2}\boldsymbol{a}^W(k) + \frac{\Delta t^3}{6}\boldsymbol{a}'(k) \\ \boldsymbol{q}\left(\boldsymbol{\omega}^W(k)\Delta t + \frac{\Delta t^2}{2}\overrightarrow{\alpha}^W(k)\right) \otimes \boldsymbol{q}^{WI}(k) \\ \boldsymbol{v}^W(k) + \boldsymbol{a}^W(k)\Delta t + \frac{\Delta t^2}{2}\boldsymbol{a}'(k) \\ \boldsymbol{\omega}^W(k) + \overrightarrow{\alpha}^W(k)\Delta t \\ \boldsymbol{a}^W(k) + \boldsymbol{a}'(k)\Delta t \\ \boldsymbol{b}^W(k) + \overrightarrow{b}'^W(k)\Delta t \end{pmatrix},$$

$$\boldsymbol{a}'(k) = \overrightarrow{j}^W(k) + \overrightarrow{\alpha}^W(k) \times \boldsymbol{v}^W(k) + \boldsymbol{\omega}^W(k) \times \boldsymbol{a}^W(k),$$

where Δt is the time step, $\boldsymbol{a}'$ is the derivative of the acceleration containing the tangential and centripetal components, $\otimes$ is the quaternion multiplication, $\boldsymbol{q}(\boldsymbol{\omega}^W(k)\Delta t + \frac{\Delta t^2}{2}\overrightarrow{\alpha}^W(k))$ is a quaternion defined by rotation $(\boldsymbol{\omega}^W(k)\Delta t + \frac{\Delta t^2}{2}\overrightarrow{\alpha}^W(k))$, and $\overrightarrow{j}^W$, $\overrightarrow{\alpha}^W$, and $\overrightarrow{b}'^W$ are the system noises. $\overrightarrow{j}^W$ is the translational jerk, $\overrightarrow{\alpha}^W$ is the angular acceleration, and $\overrightarrow{b}'^W$ is the variation of the bias of the accelerometer.

B. Measurement Model [6], [7]

The measurement vector $\boldsymbol{h}$ is:

$$\boldsymbol{h} = \begin{pmatrix} \boldsymbol{h}_q^{WI\,T} & \boldsymbol{h}_a^{W\,T} & \boldsymbol{h}_{y\alpha}^{i\;T} & \boldsymbol{h}_{y\beta}^{i\;T} & \cdots \end{pmatrix}^T,$$

TABLE I

SPECIFICATIONS OF THE ORIENTATION SENSOR

DOF	3 (Yaw, pitch, and roll)
Angular range [°]	360 (All axes)
Maximum angular rate [°/s]	1200
Minimum angular rate [°/s]	0
RMS accuracy [°/s]	Yaw: 1CPitch and roll: 0.25 (25 [°C])
RMS angular resolution [°/s]	0.03
Update rate [Hz]	30 (Maximum 180)

where $\boldsymbol{h}_q^{WI}$ is the quaternion of the measured orientation, $\boldsymbol{h}_a^{W}$ is the measured acceleration, $\boldsymbol{h}_y^{i}$ is the measured position of the feature in the camera image coordinate frame i, α and β are the ID numbers of the features measured successfully.

The measurement models of the orientation and acceleration of the interface are:

$$\begin{aligned} \boldsymbol{h}_q^{WI}(k) &= \boldsymbol{q}^{WI}(k), \\ \boldsymbol{h}_a^{W}(k) &= \boldsymbol{a}^{W}(k) + \boldsymbol{b}^{W}(k), \end{aligned}$$

and the measurement model of the image patch feature following [6] is:

$$\begin{aligned} \boldsymbol{h}_y^{i} &= \begin{pmatrix} u & v \end{pmatrix}^T = \begin{pmatrix} u_0 - f k_u \frac{y_x}{y_z} & v_0 - f k_v \frac{y_y}{y_z} \end{pmatrix}^T, \\ \boldsymbol{y}^{I} &= \begin{pmatrix} y_x & y_y & y_z \end{pmatrix}^T = \boldsymbol{R}^{IW}\left(\boldsymbol{q}^{\boldsymbol{WI}}\right)\left(\boldsymbol{y}^{W} - \boldsymbol{p}^{W}\right), \end{aligned}$$

where $\boldsymbol{y}^{I}$ is the position of the feature in the interface coordinate frame I, $\boldsymbol{R}^{IW}(\boldsymbol{q}^{\boldsymbol{WI}})$ is the rotation matrix defined by the orientation of the interface $\boldsymbol{q}^{WI}$. f, k_u, and k_v are the camera parameters; f is the focal length and k_u and k_v are the pixel densities in the horizontal and vertical directions respectively.

These models, the motion model and measurement models, are used to estimate the pose of the interface and the 3D map through the prediction and update step of EKF.

IV. EXPERIMENTS AND RESULTS

A. EXPERIMENTAL SETTING

We implemented the above algorithms on our interface (see Fig. 1) and performed SLAM and robot control experiments using it. A note PC for SLAM and robot control has a 1.33GHz Core 2 Duo processor. The resolution of the camera image is 320×240 pixels. The specifications of the orientation sensor are summarized in Table I. The accelerometer which we use is embedded in InertiaCube3. Its specification is not available; but we experimentally estimated its accuracy in advance. The interface is mounted on the right shoulder of the user as shown in the figure.

Fig. 3 shows an experimental environment, and Fig. 4 shows the world coordinate frame W of the environment. The world coordinate frame is defined as follows: the x^W axis is rightward, the y^W axis is upward, and the z^W axis is in the frontal direction from the screen. The initial user's position is the horizontal origin point in the environment. The four corners of the screen are used as prior known features. The screen is $2.0\,[\mathrm{m}]$ forward of the origin point. The size of it is $0.91\,[\mathrm{m}]$ in height by $1.22\,[\mathrm{m}]$ in width, and the center of it is at $1.5\,[\mathrm{m}]$ from the floor. Four points are at $(0.61, 0.455, 2.0)$, $(-0.61, 0.455, 2.0)$, $(-0.61, -0.455, 2.0)$, $(0.61, -0.455, 2.0)$, respectively. The object put in the environment is used to a verification of feature mapping.

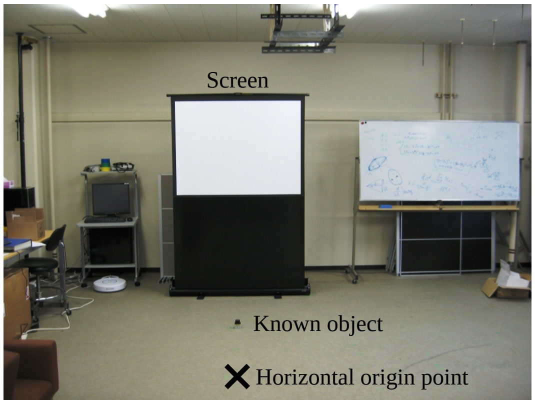


Fig. 3. Environment for the experiments.

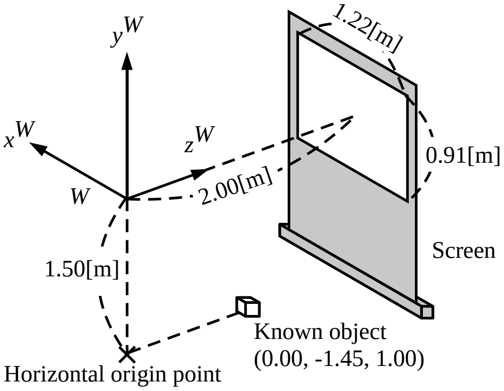


Fig. 4. World coordinate frame W of the environment.

Common parameters of SLAM are set as follows: Covariance matrices of the system noises, that is, those for the jerk $\overrightarrow{j}^W$, the angular acceleration $\overrightarrow{\alpha}^W$, and the variation of the bias of the accelerometer $\overrightarrow{b}'^W$ are respectively given by:

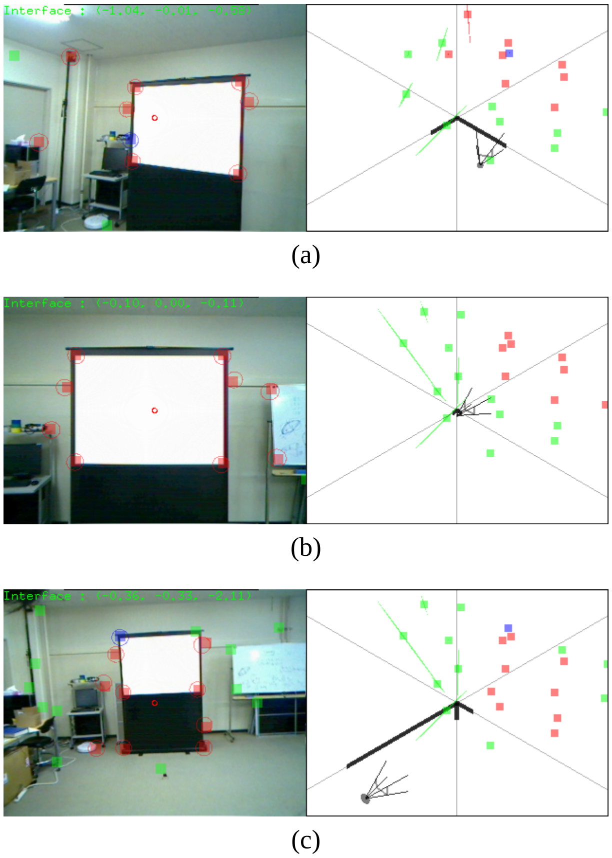
$$\boldsymbol{Q}_j = 6.0^2\boldsymbol{I}, \boldsymbol{Q}_\alpha = 9.0^2\boldsymbol{I}, \boldsymbol{Q}_{b'} = \left(1.0^{-5}\right)^2\boldsymbol{I}.$$

Covariance matrices of the measurement noises of the measured orientation of the interface $\boldsymbol{h}_q^{WI}$, the measured acceleration of the interface $\boldsymbol{h}_a^{W}$, and the feature image patch $\boldsymbol{h}_y^{i}$ is respectively:

$$\boldsymbol{R}_q = \left(3.0^{-3}\right)^2\boldsymbol{I}, \boldsymbol{R}_a = 0.5^2\boldsymbol{I}, \boldsymbol{R}_y = 1.0^2\boldsymbol{I}.$$

B. SLAM

We performed experiments of the pose estimation of the interface firstly. The experiments include a rough comparison of the estimated horizontal position of the interface (x_i^W, z_i^W) and that of the user (x_u^W, z_u^W), the evaluation of the accuracy of feature mapping using the known object, a confirmation of effectiveness of the orientation sensor and the accelerometer, and a validation of real-timeness of the method.


Interface : (-1.04, -0.01, -0.58)
Interface : (-0.10, 0.00, -0.11)
Interface : (-0.36, -0.33, -2.11)

(a)

(b)

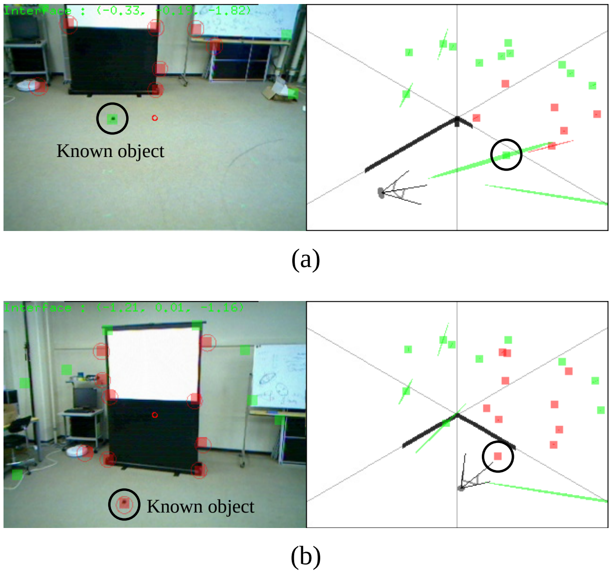
(c)

Fig. 5. Result of the pose estimation. The left column shows the camera image tracked features, and the right column shows the 3D map. (a) The 1550th frame at $(-1.00, -0.50)$. (b) The 1700th frame is at $(0.00, 0.00)$ (c) The 1850th frame is at $(0.00, -2.00)$.

TABLE II

COMPARISON OF THE USER'S POSITION AND THE ESTIMATED POSITION OF THE INTERFACE

Frame no.	User's position $[\mathrm{m}]$ (x_u^W, z_u^W)	Estimated position $[\mathrm{m}]$ (x_i^W, z_i^W)
1550	$(-1.00, -0.50)$	$(-1.04, -0.58)$
1700	$(0.00, 0.00)$	$(-0.10, -0.11)$
1850	$(0.00, -2.00)$	$(-0.36, -2.11)$


Interface : (-0.33, -0.19, -1.82)
Known object
Interface : (-1.21, 0.01, -1.16)
Known object

(a)

(b)

Fig. 6. Result of the feature mapping. (a) The 382nd frame: finish the depth initialization. (b) The 682nd frame.

In the experiment of the comparison of the positions, we define the estimated horizontal position of the interface as the horizontal position of the user's right toe. We used markers on the floor to compare the estimated and the actual position of the interface. The markers were put at $(-1.00, -0.50)$, $(0.00, 0.00)$, and $(0.00, -2.00)$.

Fig. 5 shows the pose estimation of the interface and the3D map at each marker. In this figure, the colored point or line means as follows: the red ones are measured features, the blue ones are features failing to measure, the green ones are features which the method does not try to measure since they are not in the camera image or are too far to measure, the line in the 3D map is the axes of the world coordinate frame, the ellipses in the camera image show the 3σ region defined by the innovation covariance, and the ellipsoids in the 3D map show the 3σ region defined by the covariance of the feature's position.

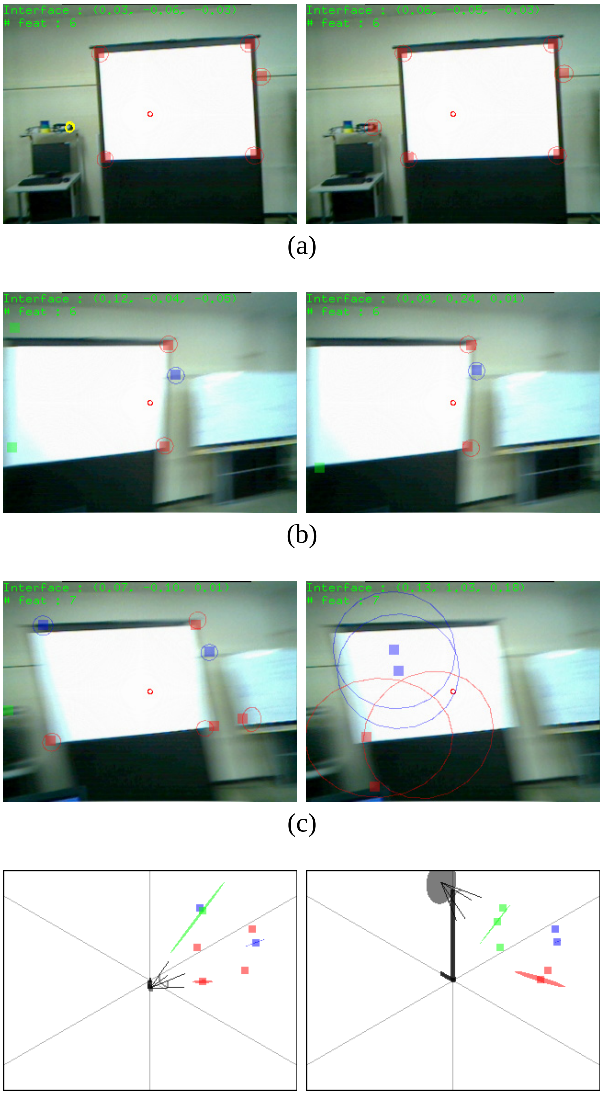
Table II shows results at each marker. In the results, the estimated positions roughly match with the corresponding use positions, but their difference is larger in the 1850th frame; the horizontal difference is about $0.38\,[\mathrm{m}]$. The reasons for the difference are as follows: the measurable features are distant and there still remains a positional difference between the user's shoulder and right toe. Although the position estimate is not very accurate, since the interface is used on-line with visual feedback with the user, this accuracy is acceptable, as shown in the robot control experiment below.

In the comparison of the feature mapping, we compared the position of the known object (Fig. 4) with its estimated position. The known object was put at $(0.00, -1.45, 1.00)$.

Fig. 6 shows the result of the feature mapping. The feature of the known object is initialized at $(-0.01, -1.66, 1.41)$ at the 382nd frame, and estimated at $(-0.07, -1.50, 1.06)$ at the 682nd frame. There is some difference between the actual position and the estimated position of the feature, and therefore the difference of the pose estimation of the interface such as Fig. 5 (c) occurs.

Next, in the confirmation of effectiveness of the orientation sensor and the accelerometer, we gave the interface a rapid motion which causes image blur, and compared the estimation with or without the orientation sensor and the accelerometer at the origin point of the environment. Fig. 7 shows the camera image of each frame and the estimated pose of the interface. The left figure of Fig. 7 (d) shows that the interface with the orientation sensor and the accelerometer can cope with a rapid motion. On the other hand, the estimation without the sensors fails. This result shows that the orientation sensor and the accelerometer are effective in rapid motions.

We then analyzed the change of the time for pose es-


(a)

(b)

(c)

(d)

Fig. 7. Rapid motion of the interface. The left column is with the orientation sensor and the accelerometer, and the right column is without them. (a) The 40th frame. (b) The 50th frame. (c) The 60th frame. (d) 3D map at the 60th frame.

timation according to the number of features used, in the case of using vision with the orientation sensor and the accelerometer. Table III shows the result. From this result, the estimation time depends on the number of the features in the 3D map. If the number of features is large, it is difficult to estimate the pose of the interface in real-time. Since a longer estimation time tends to cause errors in feature matching, it might be necessary to use only the near features to the interface for the estimation using EKF.

C. Robot Motion Control

In the experiment of robot motion control, the interface commands the robot to move and to rotate its head using the its estimated pose. The robot, Enon, has a move instruction which can uses a 2D target position in its world coordinate frame *WE* and a head rotation instruction which uses target pan and tilt angles. We set the target values as follows: the estimated horizontal position of the interface (x^W, z^W) is set to the target position of the move instruction (y^{WE}, x^{WE}) and the estimated yaw and pitch angles of the interface are set to the target pan and tilt angles of the head rotation, respectively.

TABLE III

RELATION BETWEEN THE NUMBER OF THE FEATURES IN THE 3D MAP AND THE ESTIMATION PERIOD

The number of the features	The estimation period $[\mathrm{ms}]$	The estimation frequency $[\mathrm{Hz}]$
4	33.8	29.6
10	34.5	29.0
20	35.1	28.5
30	45.7	21.9

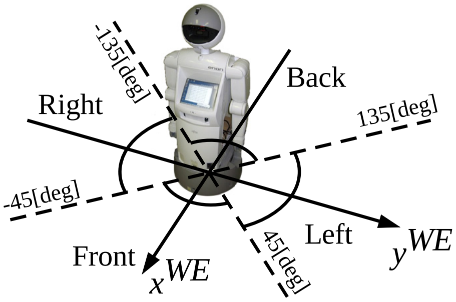


Fig. 8. Definition of the robot direction.

Fig. 8 shows the definition of the longitudinal and the lateral direction with respect to the robot. The interface sends move commands every about one second (corresponding to 32 frames) and rotation commands every about 0.5 seconds (corresponding to 16 frames). Since it is not desirable to repeatedly give commands for small movements to the robot, and since the robot cannot move instantaneously laterally due to its two-wheel drive mechanism, the interface sends move commands when the estimated moving distance of the interface is larger than or equal to $0.2\,[\mathrm{m}]$ and $0.8\,[\mathrm{m}]$ in the longitudinal and the lateral direction, respectively. The maximum speed of the robot motion is set to $0.3\,[\mathrm{m/s}]$. Concerning the rotation of the head, since the movable range of the head is limited, if the estimated interface orientation is out of range, the target rotation angles are determined in the movable range which is nearest to the estimated orientation. Fig. 9 shows the trajectory of the user's motion and the initial robot position. The origin of the robot's world coordinate frame *WE* has an offset of $1.5\,[\mathrm{m}]$ on the right of that of the user's (i.e., the interface's) one W.

Fig. 10 shows the experimental result. The initial position of (a) and the final position of (j) are about the same position. Although there is currently a time delay between the pose estimation and the execution of motion commands in the robot, the robot was able to follow the user's movement reasonably well. This result shows the effectiveness of the interface as the tool for instructing robot motion.

V. CONCLUSIONS AND FUTURE WORK

In this paper we have developed an interface for controlling robot motion easily. Using an orientation sensor and an accelerometer in addition to vision, the interface can cope with a relatively fast motion for which the estimation by only vision certainly fails. In the experiment of robot motion control, although there is sometimes a few second delay due to some system limitation, the robot can reach the target pose determined by the interface. The accuracy of pose estimation and the estimation time are acceptable for the preliminary robot control experiments, but more improvements are needed to cope with more rapid and complex motions.

This system has two advantages; it does not require any environmental settings except a few known feature points and does not restrict the user's field of activities because it is a wearable system using no environmental sensors.

This paper focused on the pose estimation of the interface and its use for robot motion control, and does not address the human-robot bidirectional interaction. Gesture-based interaction or sharing the image data between the robot and the user are possible extensions for this purpose. It might also be useful to instruct the robot how to move the hands by estimating the user's hand motions.

ACKNOWLEDGMENT

This work is in part supported by IJARC CORE Project.

REFERENCES

[1] S. B. Kang and K. Ikeuchi, "Toward automatic robot instruction from perception – temporal segmentation of tasks from human hand motion," *IEEE Trans. on Robotics and Automation*, vol. 11, no. 5, pp. 670–681, 1995.

[2] M. Ehrenmann, O. Rogalla, R. Zöllner, and R. Dillmann, "Teaching service robots complex tasks: Programming by demonstration for workshop and household environments," in *Proc. of 2001 Int. Conf. on Field and Service Robots*, 2001, pp. 397–402.

[3] M. Kourogi and T. Kurata, "A method of personal positioning based on sensor data fusion of wearable camera and self-contained sensors," in *Proc. of IEEE Int. Conf. on Multisensor Fusion and Integration for Intelligent Systems*, 2003, pp. 287–292.

[4] T. Harada, T. Gyota, Y. Kuniyoshi, and T. Sato, "Development of wireless networked tiny orientation device for wearable motion capture and measurement of walking around, walking up and down, and jumping tasks," in *Proc. of IEEE/RSJ 2007 Int. Conf. on Intelligent Robots and Systems*, 2007, pp. 4135–4140.

[5] T. Maeda, H. Ando, M. Sugimoto, J. Watanabe, and T. Miki, "Wearable robotics as a behavioral interface – the study of the parasitic humanoid –," in *Proc. of 6th Int. Symp. on Wearable Computers*, 2002, pp. 145–151.

[6] A. J. Davison, W. Mayol, and D. W. Murray, "Real-time localisation and mapping with wearable active vision," in *Proc. of the 2nd IEEE/ACM Int. Symp. on Mixed and Augmented Reality*, 2003, pp. 18–27.

[7] L. Armesto, J. Tornero, and M. Vincze, "Fast ego-motion estimation with multi-rate fusion of inertial and vision," *The International Journal of Robotics Research*, vol. 26, no. 6, pp. 577–589, 2007.

[8] J. Shi and C. Tomasi, "Good features to track," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 1994, pp. 593–600.

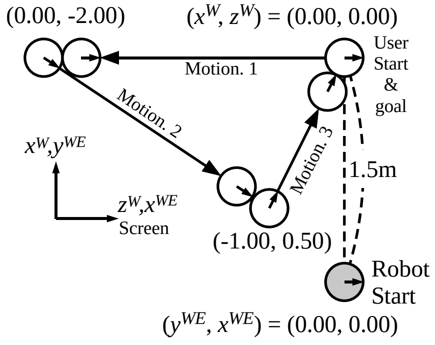


Fig. 9. Trajectory of the user's motion and the initial robot pose.

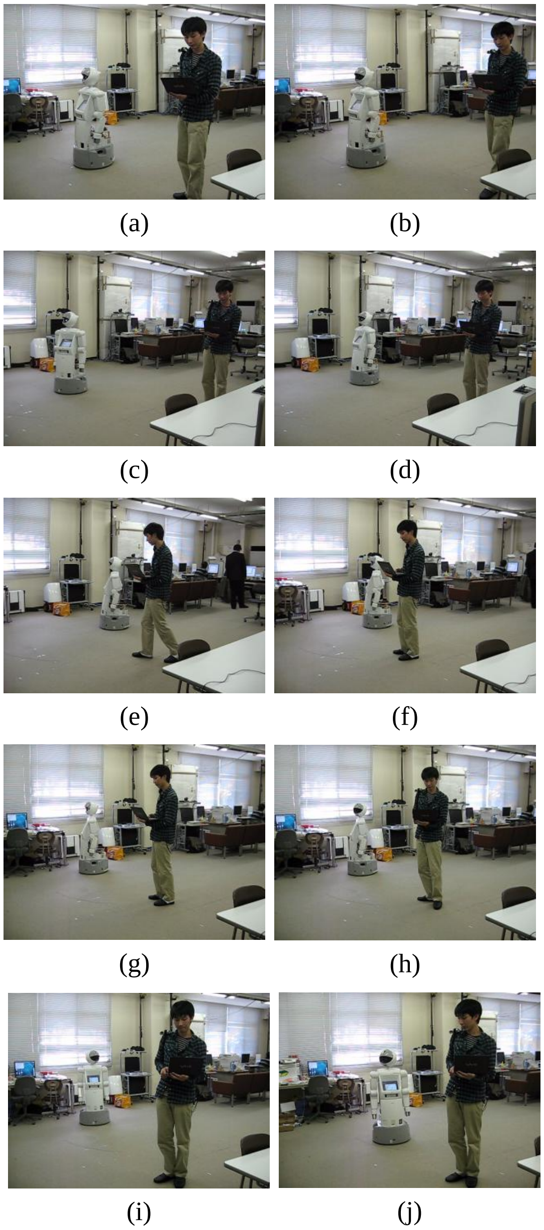

(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Fig. 10. Experiment of the robot motion control. (a)–(d): Motion. 1 (0–9 [s]); the user moves backward and the robot follows the motion. (e)–(g): Motion. 2 (13–20 [s]); the user moves diagonally forward right and the robot follows. (h)–(j): Moiton. 3 (23–31 [s]); the user moves diagonally forward left and the robot follows.