Proceedings of the 2011 IEEE
International Conference on Robotics and Biomimetics
December 7-11, 2011, Phuket, Thailand

A Wearable Robot Control Interface based on Measurement of Human Body Motion using a Camera and Inertial Sensors

Junichi Sugiyama and Jun Miura

Abstract— **This paper describes a wearable robot control interface based on measurement of human body motion, which allows a user to control the robot intuitively. This wearable interface is composed of a camera and inertial sensors and estimates body motion of the user: movement and arm motion. The estimated motion is then converted into commands of movement or arm motion of a humanoid robot. The body motion estimation is performed by a combination of a monocular SLAM and an EKF-based motion estimation using inertial sensors. The arm motion is estimated with a Wii Remote with a visual marker using the same motion model and sensor data integration. We implemented efficient algorithms, especially for image processing, for a real-time motion estimation. We conducted motion estimation and robot control experiments, which show the effectiveness of the proposed method.**

I. INTRODUCTION

Service robots are expected to help human in various everyday situations, such as bringing a user-specified object, cleaning a room, mobile aid, and social interaction. However it is difficult to give a robot a complete set of required skills and knowledge for various tasks in advance. Therefore on-site teaching is important and necessary for such robots.

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 propose a method to instruct movement of a humanoid robot intuitively using human body movement as well as arm motion.

In such a method, it is necessary to measure a human motion and give it as the target motion to the robot. Several systems for pose (position and orientation) estimation have been developed. Matsuyama *et al.* [3] developed a system using several environmental cameras. However in such a system its measurable area is limited by the field of view of the system.

Harada *et al.* [4] developed a system for recognizing human motion. They attach sensor modules including inertial sensors to the lower limbs to measure their motion thereby recognizing the motion type such as walking and jumping. 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. In such a system using inertial sensors, there is a problem caused by a drift of inertial sensors. In addition, it needs the time for wearing a lot of sensors.

Our pose estimation method uses a few wearable sensors, a camera and inertial sensors. The information from these sensors is used to estimate body movement and arm motion of a user. The body movement estimation is based on a SLAM (simultaneously localization and mapping) using a camera and inertial sensors which are fused by extended Kalman filter (EKF) [6]. The arm motion estimation is based on the same framework as the body movement estimation, that is vision-inertial fusion with EKF. The estimated body movement and arm motion is used to issue a command for controlling a humanoid robot. We implemented an experimental system 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 and IV explain the pose estimation algorithm of the body and the arm using EKF, respectively. Sec. V describes experimental results on the pose estimation and robot control. Sec. VI concludes the paper and discusses future work.

II. SYSTEM OVERVIEW

Fig. 1 shows a user with the interface which we developed. The interface is composed of a head-mounted part and a Wii Remote with a Wii Motion Plus. The head-mounted part has a camera (ARTCAM-022MINI by ARTRAY) and a 3-axis orientation sensor (InertiaCube3 by InterSense) attached to a monocular head-mounted display. The orientation sensor includes a 3-axis accelerometer, a 3-axis gyroscope, and a 3-axis magnetometer. The Wii Remote has a red-colored visual marker for image measurement and is held by the hand of the user.

The data from these sensors are processed with a laptop PC for estimation of the movement and arm motion of the user. The movement estimation is performed by a SLAM using EKF. It uses the camera image and the inertial data from the accelerometer and the orientation sensor integrated in the InertiaCube3. The arm motion is estimated by using EKF with the same motion model as the movement estimation. It uses the camera image and the inertial data from the accelerometer and the gyroscope integrated in Wii Remote.

Fig. 2 is the humanoid robot, Enon by Fujitsu, which has a wheeled platform, a rotatable head, and arms. The estimated movement and arm motion are used to issue a command for control this humanoid robot.

J. Sugiyama is with Department of Computer Science and Engineering, Toyohashi University of Technology, Toyohashi, Aichi, 441-8580, Japan. sugiyama@aisl.cs.tut.ac.jp

J. Miura is with Department of Computer Science and Engineering, Toyohashi University of Technology, Toyohashi, Aichi, 441-8580, Japan. jun.miura@tut.jp

978-1-4577-2137-3/11/$26.00 © 2011 IEEE

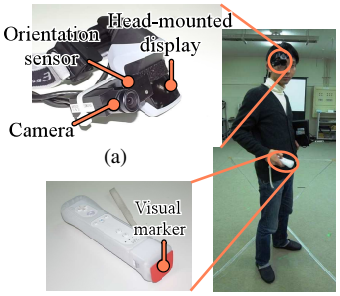


Fig. 1. User with the interface. (a) Head-mounted part which has a camera and an orientation sensor; (b) Wii Remote with a red-colored visual marker.

Fig. 2. Humanoid robot, Enon by Fujitsu.

III. MOVEMENT ESTIMATION

The movement estimation method is based on a monocular SLAM with sensor fusion of vision and inertial using EKF which we have developed [6]. This method is based on a monocular SLAM using EKF developed by Davison *et al.* [7] and a localization method with sensor fusion of vision and inertial using EKF developed by Armesto *et al.* [8].

The SLAM method [7] uses only a monocular camera and estimates 3D pose (position and orientation) of the camera and the 3D map of image features around the camera. The state vector to estimate includes the pose of the 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. The state initialization method of a newly detected feature limits the depth of the feature. Therefore in this research, we use a monocular SLAM with inverse depth parameterization proposed by Montiel *et al.* [9].

In this parameterization, a 3D position of image feature is represented with the 3D position of the camera at the time of first detection $(\ x_0 \quad y_0 \quad z_0\ )$, the 2D direction $(\ \theta \quad \phi\ )$, and the inverse depth of the feature ρ; $\boldsymbol{y} = (\ x_0 \quad y_0 \quad z_0 \quad \theta \quad \phi \quad \rho\ )$.

It can represent the feature at infinity by setting the inverse depth to zero. The image feature matching as an observation is performed in limited region for real-time estimation, and therefore the matching fails if the camera move too fast.

To cope with this problem, we additionally use inertial sensors. The localization method [8] estimates ego-motion by integrating data from a camera and inertial sensors using EKF. The state vector to estimate includes the pose of the robot, the translational and rotational velocities, the translational acceleration, and the bias of the accelerometer. The information of the camera image can compensate a drift caused by inertial sensors, and the inertial data can coop with a rapid motion which causes image blur. We combine the above two methods for a reliable movement estimation in a complex environment.

A. Motion Model [6]

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

$$\boldsymbol{x} = \begin{pmatrix} \boldsymbol{x}_I^W \\ \boldsymbol{y}_1^W \\ \boldsymbol{y}_2^W \\ \vdots \end{pmatrix}, \boldsymbol{P} = \begin{bmatrix} \boldsymbol{P}_{x_I x_I} & \boldsymbol{P}_{x_I y_1} & \boldsymbol{P}_{x_I y_2} & \cdots \\ \boldsymbol{P}_{y_1 x_I} & \boldsymbol{P}_{y_1 y_1} & \boldsymbol{P}_{y_1 y_2} & \cdots \\ \boldsymbol{P}_{y_2 x_I} & \boldsymbol{P}_{y_2 y_1} & \boldsymbol{P}_{y_2 y_2} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}, \quad (1)$$

where $\boldsymbol{x}_I$ is the estimated state vector of the interface, $\boldsymbol{y}_i (i = 1, 2, \cdots)$ is the estimated position of a feature, and superscript W means a vector in the world coordinate frame W. The state vector of the interface $\boldsymbol{x}_I$ is:

$$\boldsymbol{x}_I^W = (\ \boldsymbol{p}_I^{WT} \quad \boldsymbol{q}_I^{WT} \quad \boldsymbol{v}_I^{WT} \quad \boldsymbol{\omega}_I^{WT} \quad \boldsymbol{a}_I^{WT} \quad \boldsymbol{b}_I^{WT}\ )^T, \quad (2)$$

where $\boldsymbol{p}_I$, $\boldsymbol{v}_I$, and $\boldsymbol{a}_I$ are respectively the position, the velocity, and the acceleration of translation of the interface, and $\boldsymbol{q}_I$ and $\boldsymbol{\omega}_I$ are respectively the orientation and the angular velocity of the interface. $\boldsymbol{q}$ is represented by quaternion and $\boldsymbol{\omega}_I$ declares that its norm is a rotation angle and its orientation is a rotation axis. $\boldsymbol{b}_I$ is the bias of the accelerometer [8].

The motion model of the interface is:

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

$$\boldsymbol{a}'(k) = \vec{j}_I^W(k) + \vec{\alpha}_I^W(k) \times \boldsymbol{v}_I^W(k) + \boldsymbol{\omega}_I^W(k) \times \boldsymbol{a}_I^W(k), \quad (4)$$

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, $\boldsymbol{q}(\boldsymbol{\omega}_I^W(k)\Delta t + \Delta t^2 \vec{\alpha}_I^W(k)/2)$ is a quaternion defined by rotation $(\boldsymbol{\omega}_I^W(k)\Delta t + \Delta t^2 \vec{\alpha}_I^W(k)/2)$, and $\vec{j}_I^W$, $\vec{\alpha}_I^W$, and $\vec{b}_I'^W$ are the system noises. $\vec{j}_I^W$ is the translational jerk, $\vec{\alpha}_I^W$ is the angular acceleration, and $\vec{b}_I'^W$ is the variation of the bias of the accelerometer. The covariance matrices of these noises are $\boldsymbol{Q}_{\vec{j}_I} = 15^2\boldsymbol{I}$ ($\sigma_{\vec{j}_I} = 15\,[\mathrm{m/s^3}]$), $\boldsymbol{Q}_{\vec{\alpha}_I} = 15^2\boldsymbol{I}$ ($\sigma_{\vec{\alpha}_I} = 15\,[\mathrm{rad/s^2}]$), and $\boldsymbol{Q}_{\vec{b}_I'} = (10^{-3})^2\boldsymbol{I}$ ($\sigma_{\vec{b}_I'} = 10^{-3}\,[\mathrm{m/s^3}]$), respectively.

B. Measurement Model [6]

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

$$\boldsymbol{h} = (\ \boldsymbol{h}_{q_I}^{WT} \quad \boldsymbol{h}_{a_I}^{WT} \quad \boldsymbol{h}_{yn}^{iT} \quad \boldsymbol{h}_{ym}^{iT} \quad \cdots\ )^T, \quad (5)$$

where $\boldsymbol{h}_{q_I}^W$ is the quaternion of the measured orientation, $\boldsymbol{h}_{a_I}^W$ is the measured acceleration, $\boldsymbol{h}_y^i$ is the measured position of the feature in the camera image coordinate frame i, n and m are the ID numbers of the features measured successfully.

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

$$\boldsymbol{h}_{q_I}^W(k) = \boldsymbol{q}_I^W(k), \quad (6)$$

$$\boldsymbol{h}_{a_I}^W(k) = \boldsymbol{a}_I^W(k) + \boldsymbol{b}_I^W(k), \quad (7)$$

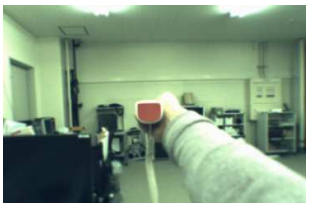


Fig. 3. Wii Remote with the visual marker in the camera image.

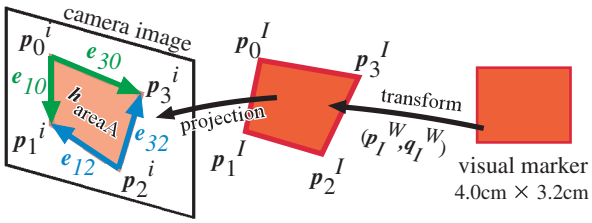


Fig. 4. Relation of the vectors of the measurement model of $\boldsymbol{h}^i_{areaA}$.

and the measurement model of the feature $\boldsymbol{y}^W$ [9] is:

$$\boldsymbol{h}^i_y = \begin{pmatrix} u & v \end{pmatrix}^T = \begin{pmatrix} u_0 - fk_u \frac{y^I_x}{y^I_z} & v_0 - fk_v \frac{y^I_y}{y^I_z} \end{pmatrix}^T, \quad (8)$$

$$\boldsymbol{y}^I = \begin{pmatrix} y^I_x \\ y^I_y \\ y^I_z \end{pmatrix} = \boldsymbol{R}(\bar{\boldsymbol{q}}^W_I) \left\{ \rho \left(\begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix} - \boldsymbol{p}^W_I \right) + \boldsymbol{m}(\theta, \phi) \right\}, \quad (9)$$

where $\boldsymbol{y}^I$ is the position of the feature in the interface coordinate frame I corresponding with the camera coordinate frame of the head-mounted part, $\boldsymbol{R}(\bar{\boldsymbol{q}}^W_I)$ is the rotation matrix defined by the inverse quaternion of the interface $\boldsymbol{q}^W_I$, and $\boldsymbol{m}(\theta, \phi)$ is the directional vector defined by θ and ϕ. 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.

The covariance matrices of these models are $\boldsymbol{R}_{\boldsymbol{h}_{q_I}} = (10^{-3})^2 \boldsymbol{I}$ ($\sigma_{\boldsymbol{h}_{q_I}} = 10^{-3}$) for $\boldsymbol{h}^W_{q_I}$, $\boldsymbol{R}_{\boldsymbol{h}_{a_I}} = (0.5)^2 \boldsymbol{I}$ ($\sigma_{\boldsymbol{h}_{a_I}} = 0.5\,[\mathrm{m/s^2}]$) for $\boldsymbol{h}^W_{a_I}$, and $\boldsymbol{R}_{\boldsymbol{h}^i_y} = 1^2 \boldsymbol{I}$ ($\sigma_{\boldsymbol{h}^i_y} = 1\,[\mathrm{pixel}]$) for $\boldsymbol{h}^i_y$.

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.

C. Implementation

To accelerate the speed of the movement estimation, several image processing routines have been developed using Intel SSE instructions. This SIMD instructions can process 16 bytes-aligned data composed of the same data type at the same time. We use the instructions for image undistortion, gray-scale transformation, and image feature detection and matching. For example, in the matching, the patches of the detected feature and the camera image to calculate matching are copied to 16 bytes-aligned data, and then calculate the sum of the absolute differences (SAD) using the SAD instruction of SSE. This matching using SSE is about 15 times faster than simple C/C++ implementation; $156\,[\mu\mathrm{s}]$ for SSE and $2.40\,[\mathrm{ms}]$ for C/C++.

IV. ARM MOTION ESTIMATION

The arm motion is estimated from the information of the visual marker of the Wii Remote (Fig. 1(b)) in the camera image and the inertial data of the accelerometer and the gyroscope of the Wii Remote. Fig. 3 shows the camera image with visual marker of the Wii Remote. In the current implementation, it is required to direct the Wii Remote to front at the beginning of the arm motion estimation as shown in Fig. 3, because the yaw position of the Wii Remote cannot be recognized from the visual marker or the inertial data.

A. Motion Model

The arm motion estimation uses the same motion model as the movement estimation. The following is the correspondence of the state vectors to be estimated between the movement and the arm motion:

$$\boldsymbol{x}^I_A = \begin{pmatrix} \boldsymbol{p}^{IT}_A & \boldsymbol{q}^{IT}_A & \boldsymbol{v}^{IT}_A & \boldsymbol{\omega}^{IT}_A & \boldsymbol{a}^{IT}_A & \boldsymbol{b}^{IT}_A \end{pmatrix}^T$$
$$\Rightarrow \boldsymbol{x}^W_I = \begin{pmatrix} \boldsymbol{p}^{WT}_I & \boldsymbol{q}^{WT}_I & \boldsymbol{v}^{WT}_I & \boldsymbol{\omega}^{WT}_I & \boldsymbol{a}^{WT}_I & \boldsymbol{b}^{WT}_I \end{pmatrix}^T, \quad (10)$$

where $\boldsymbol{x}^I_A$ is the state vector of the arm in the interface (camera) coordinate frame I. In the arm motion estimation, the arm state is estimated in I.

The motion model is the same model as (3) under the above correspondence.

B. Measurement Model

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

$$\boldsymbol{h}_A = \begin{pmatrix} \boldsymbol{h}^{iT}_{pA} & \boldsymbol{h}^{iT}_{areaA} & \boldsymbol{h}^{IT}_{\omega A} & \boldsymbol{h}^{IT}_{PRA} \end{pmatrix}^T, \quad (11)$$

where $\boldsymbol{h}^i_{pA}$ and $\boldsymbol{h}^i_{areaA}$ is the position and the area of the visual marker in the image coordinate frame i, respectively; $\boldsymbol{h}^I_{\omega A}$ is the angular velocity in the interface coordinate frame I; $\boldsymbol{h}^I_{PRA}$ is the pitch and roll angles of the Wii Remote.

The model of the $\boldsymbol{h}^i_{pA}$ is:

$$\boldsymbol{h}^i_{pA} = \begin{pmatrix} u_{pA} \\ v_{pA} \end{pmatrix} = \begin{pmatrix} u_0 - fk_u \frac{p^I_{Ax}}{p^I_{Az}} \\ v_0 - fk_v \frac{p^I_{Ay}}{p^I_{Az}} \end{pmatrix}, \quad (12)$$

where p^I_{Ax}, p^I_{Ay}, and p^I_{Az} are the element of $\boldsymbol{p}^I_A$. The covariance matrix of $\boldsymbol{h}^i_{pA}$ is $R_{\boldsymbol{h}_{pA}} = (10/3)^2 \boldsymbol{I}$ ($\sigma_{\boldsymbol{h}_{pA}} = 10/3\,[\mathrm{pixel}]$).

The model of the $\boldsymbol{h}^i_{areaA}$ is:

$$\boldsymbol{h}^i_{areaA} = \boldsymbol{H}_{area}(\boldsymbol{e}_{10}, \boldsymbol{e}_{30}, \boldsymbol{e}_{12}, \boldsymbol{e}_{13})$$
$$= \left(\sqrt{(x_{10x} y_{30y} - y_{10} x_{30})^2} + \sqrt{(x_{12} y_{32} - y_{12} x_{32})^2} \right) / 2, \quad (13)$$

$$\boldsymbol{e}_{10} = \begin{pmatrix} x_{10} & y_{10} \end{pmatrix}^T = \boldsymbol{p}^i_1 - \boldsymbol{p}^i_0, \quad \boldsymbol{e}_{30} = \begin{pmatrix} x_{30} & y_{30} \end{pmatrix}^T = \boldsymbol{p}^i_3 - \boldsymbol{p}^i_0,$$
$$\boldsymbol{e}_{12} = \begin{pmatrix} x_{12} & y_{12} \end{pmatrix}^T = \boldsymbol{p}^i_1 - \boldsymbol{p}^i_2, \quad \boldsymbol{e}_{32} = \begin{pmatrix} x_{32} & y_{32} \end{pmatrix}^T = \boldsymbol{p}^i_3 - \boldsymbol{p}^i_2, \quad (14)$$

where $\boldsymbol{p}^i_k (k = 0, 1, 2, 3)$ is the position of each projected vertex of the marker in the image coordinate frame. $\boldsymbol{e}_{10}$ and $\boldsymbol{e}_{30}$ are the vectors from $\boldsymbol{e}^i_0$ to $\boldsymbol{e}^i_1$ and $\boldsymbol{e}^i_3$, respectively; $\boldsymbol{e}_{12}$ and $\boldsymbol{e}_{32}$ are the vectors from $\boldsymbol{e}^i_2$ to $\boldsymbol{e}^i_1$ and $\boldsymbol{e}^i_3$, respectively. Fig. 4 shows this relation of these vectors. The covariance matrix of $\boldsymbol{h}^i_{areaA}$ is $\boldsymbol{R}_{\boldsymbol{h}_{areaA}} = (0.3 Z_{areaA})^2 \boldsymbol{I}$ ($\sigma_{\boldsymbol{h}_{areaA}} = 0.3 Z_{areaA}\,[\mathrm{pixel}^2]$), where Z_{areaA} is the measured area of the visual marker in the camera image.

To calculate the actual measurements, the position $\boldsymbol{z}_{pA}$ and the area Z_{areaA} of the marker, the marker region is extracted by hue extraction (0° to 20°), labeling, and extraction of the labeled region that has the maximum area. From the extracted marker region, the centroid and the area are calculated and then used as $\boldsymbol{z}_{pA}$ and Z_{areaA} respectively.

The model of $\boldsymbol{h}^I_{\omega A}$ is simply described as:

$$\boldsymbol{h}^I_{\omega A} = \boldsymbol{\omega}^I_A. \tag{15}$$

The covariance of $\boldsymbol{h}^I_{\omega A}$ is $\boldsymbol{R}_{\boldsymbol{h}_{\omega A}} = (0.01)^2\boldsymbol{I}$ ($\sigma_{\omega A} = 0.01\,[\mathrm{rad/s}]$). The actual measurement $z^I_{\omega A}$ corresponding with this model is affected by the body movement. Therefore the elimination of the movement is required:

$$\boldsymbol{q}(\boldsymbol{z}_{\omega A+I}\Delta t) \overset{\otimes\bar{\boldsymbol{q}}^I_A}{\rightarrow} \boldsymbol{q}^I_{A+I} \overset{\otimes\bar{\boldsymbol{q}}^W_I}{\rightarrow} \boldsymbol{q}^W_{A+I} \overset{\otimes\Delta\boldsymbol{q}^W_I}{\rightarrow} \boldsymbol{q}^W_A \overset{\otimes\boldsymbol{q}^W_I}{\rightarrow} \boldsymbol{q}^I_A \overset{\otimes\boldsymbol{q}^I_A}{\rightarrow} \boldsymbol{q}_A, \tag{16}$$

$$\boldsymbol{q}_A = \begin{pmatrix} q_{rA} & q_{xA} & q_{yA} & q_{zA} \end{pmatrix}^T, \tag{17}$$

$$\boldsymbol{z}^I_{\omega A} = \frac{2\arccos(q_{rA})}{\Delta t\sqrt{q^2_{xA}+q^2_{yA}+q^2_{zA}}} \begin{pmatrix} q_{xA} & q_{yA} & q_{zA} \end{pmatrix}^T, \tag{18}$$

where $\boldsymbol{q}(\boldsymbol{z}_{\omega A+I}\Delta t)$ is the amount of the rotation defined by the time step Δt and the raw angular velocity measurement of the Wii Remote $\boldsymbol{z}_{\omega A+I}$ that includes the arm and the body motion; $\Delta\boldsymbol{q}^W_I$ is the difference of the estimated orientation of the interface between $\boldsymbol{q}^W_I(k)$ and $\boldsymbol{q}^W_I(k-1)$; and $\boldsymbol{q}^I_A$ is the rotation of the arm in I. (16) shows the coordinate transform and the elimination of the body (interface) movement $\Delta\boldsymbol{q}^W_I$ and (18) shows the conversion from quaternion to angular velocity. This processing realizes simultaneous body-arm motion estimation.

The model of the pitch and roll angles $\boldsymbol{h}^I_{PRA}$ is:

$$\boldsymbol{h}^I_{PRA} = \begin{pmatrix} h^I_{pitchA} \\ h^I_{rollA} \end{pmatrix} = \begin{pmatrix} -\arctan\left(y_{PA}/\sqrt{x^2_{PA}+z^2_{PA}}\right) \\ \arctan\left(y_{RA}/\sqrt{x^2_{RA}+z^2_{RA}}\right) \end{pmatrix}, \tag{19}$$

$$\boldsymbol{v}^I_{pitchA} = \begin{pmatrix} x_{PA} & y_{PA} & z_{PA} \end{pmatrix}^T = \boldsymbol{R}(\boldsymbol{q}^I_A)\begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^T, \tag{20}$$

$$\boldsymbol{v}^I_{rollA} = \begin{pmatrix} x_{RA} & y_{RA} & z_{RA} \end{pmatrix}^T = \boldsymbol{R}(\boldsymbol{q}^I_A)\begin{pmatrix} 1 & 0 & 0 \end{pmatrix}^T, \tag{21}$$

where $\boldsymbol{v}^I_{pitchA}$ and $\boldsymbol{v}^I_{rollA}$ are the vectors defined by the orientation of the arm $\boldsymbol{q}^I_A$. The covariance matrix of these models is $\boldsymbol{R}_{\boldsymbol{h}_{PRA}} = (0.5^\circ)^2\boldsymbol{I}$, ($\sigma_{\boldsymbol{h}_{PRA}} = 0.5\,[^\circ]$, in the actual implementation, radian representation is used). The actual measurements of the pitch and the roll angle of the Wii Remote are calculated from its acceleration. Therefore these measurements are used only when the measured acceleration of the Wii Remote in I, $\boldsymbol{a}^I_{wii}$, is nearly equal to the gravitational acceleration in I, $\boldsymbol{g}^I$; if $|\boldsymbol{a}^I_{wii} - \boldsymbol{g}^I| \leq 0.02$ then these measurements are valid.

The actual measurement of the pitch and roll angles, $\boldsymbol{z}^I_{PRA}$, are also affected by the body movement, that is the pose of the interface $\boldsymbol{q}^W_I$:

$$\boldsymbol{z}^I_{PRA} = \boldsymbol{z}_{wii} - \begin{pmatrix} -\arctan\left(y^W_{PI}/\sqrt{x^{W2}_{PI}+z^{W2}_{PI}}\right) \\ \arctan\left(y^W_{RI}/\sqrt{x^{W2}_{RI}+z^{W2}_{RI}}\right) \end{pmatrix}, \tag{22}$$

$$\boldsymbol{v}^W_{pitchI} = \begin{pmatrix} x_{PI} & y_{PI} & z_{PI} \end{pmatrix}^T = \boldsymbol{R}(\boldsymbol{q}^W_I)\begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^T, \tag{23}$$

$$\boldsymbol{v}^W_{rollI} = \begin{pmatrix} x_{RI} & y_{RI} & z_{RI} \end{pmatrix}^T = \boldsymbol{R}(\boldsymbol{q}^W_I)\begin{pmatrix} 1 & 0 & 0 \end{pmatrix}^T, \tag{24}$$

where $\boldsymbol{z}_{wii}$ is the vector of the pitch and roll angles of the Wii Remote and $\boldsymbol{v}^W_{pitchI}$ and $\boldsymbol{v}^W_{rollI}$ are the vectors defined by the orientation of the body $\boldsymbol{q}^W_I$.

TABLE I
SPECIFICATIONS OF THE INERTIA CUBE 3

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

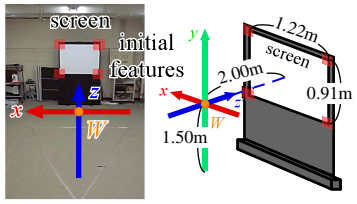


Fig. 5. Environment for the experiments and the world coordinate frame W.

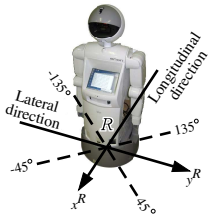


Fig. 6. Robot coordinate frame R.

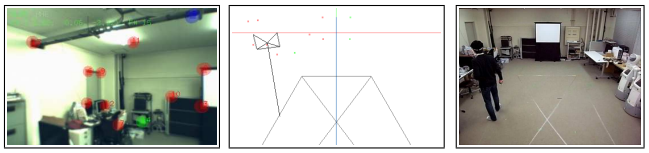
V. EXPERIMENTS AND RESULTS

We implemented the above algorithms on our interface and performed SLAM, arm motion estimation, and robot control experiments. A laptop PC for the estimation and the robot control has a 1.33GHz Core 2 Duo processor. The resolution of the camera image is 368×240 pixels. The specifications of the InertiaCube3 on the head-mounted part are summarized in Table I. In this table, the specification of the accelerometer embedded in the InertiaCube3 is not listed, because it is unavailable. The specifications of the Wii Remote are also unavailable; therefore we experimentally estimated its accuracy in advance. Fig. 5 shows an experimental environment and the world coordinate frame W and Fig. 6 shows the robot coordinate frame R. The initial position of the user is the horizontal point $(x^W, z^W) = (0, -2.5)$ in the environment. The four corners of the screen in the environment are used as prior known features for the movement estimation. The screen is $2.0\,[\mathrm{m}]$ forward of the origin point. Its size is $0.91\,[\mathrm{m}]$ in height and $1.22\,[\mathrm{m}]$ in width, and its center is at $1.5\,[\mathrm{m}]$ from the floor. These 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.

A. Result of Movement Estimation

We performed experiments of the movement estimation firstly. The experiments include a rough comparison of the estimated horizontal position of the interface (x^W_I, z^W_I) and that of the user (x^W_u, z^W_u), and a confirmation of effectiveness of the vision-inertial sensor fusion.

Fig. 7 shows an example of the camera frame and the results. In this figure, squares and ellipses indicate image features and their uncertainties, respectively. The color of red, blue, and green mean the measured feature, the feature failed to measure, and the feature that is not selected to measure, respectively.

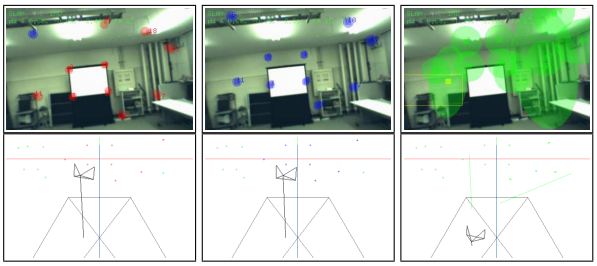


(a) (b) (c)

Fig. 7. Example of the camera frame and the results. (a): the camera frame of the interface. (b): the estimated pose and map. (c): the experimental scene.

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

User's position [m] (x_u^W, z_u^W)	Estimated position [m] (x_I^W, z_I^W)	Error [m]
$(1.25, -2.50)$	$(1.22, -2.32)$	0.18
$(1.25, 0.00)$	$(1.31, 0.07)$	0.09
$(0.00, -2.50)$	$(0.19, -2.43)$	0.20
$(-1.25, 0.00)$	$(-1.03, -0.09)$	0.24



(a) (b) (c)

Fig. 8. Results of the movement estimation with or without the vision-inertial sensor fusion. (a) is the result with the sensor fusion; (b) with only the camera; and (c) with only the inertial sensors.

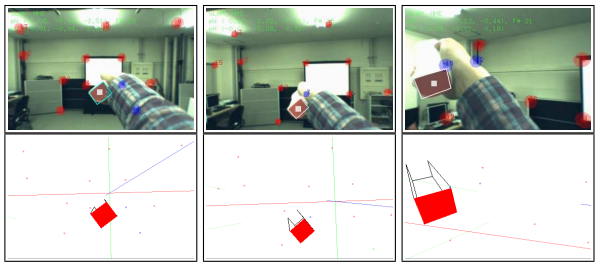
Table II shows the comparison between the estimated position and the user's. In the results, the estimated positions roughly match with the corresponding user positions; the maximum error is about $25\,[\mathrm{cm}]$.

Fig. 8 shows the effectiveness of the sensor fusion. In this experiment, the user jumped at the initial position. With the sensor fusion (Fig. 8(a)), the movement is estimated successfully. However with only the camera (Fig. 8(b)), all features fail to match (blue colored points) in the camera frame. With only the inertial sensors (Fig. 8(c)), the estimated pose shown in the map is not correct because the drift of the sensors are accumulated.

The average cycle time of the estimation depends on the number of the feature in the map: $36.7\,[\mathrm{ms}]$ (nearly equal to the frame rate of the camera) for 20 features and $43.3\,[\mathrm{ms}]$ for 40 features. This time is shorter than that of our previous method [6] in spite of increasing in the dimension per feature. This result is achieved by the developed fast algorithms.

B. Results of Arm Motion Estimation

Next, we confirmed the performance of the arm motion estimation. In this experiment, the arm motion and the movement of the user were estimated simultaneously. Fig. 9 shows the estimation result when the user moved the right arm with the Wii Remote and walked diagonally forward left at the same time. This result shows that the estimated pose of



(a) (b) (c)

Fig. 9. Results of the simultaneous movement and arm motion estimation. The upper row is the camera frame, and the lower is the estimated pose of the Wii Remote. (a) is the 1284th frame; (b) is the 1363rd; and (c) is the 1510th.

the Wii Remote is close to the actual pose even if the user's body moves. However in the current implementation we do not consider the problem that the features for the movement estimation incorrectly match with those on the arm in the camera image. In order to solve this problem, it might be necessary to use only the features lying away from the arm region.

In the case of stretching the arm forward, as shown in Fig. 3, the difference between the estimated depth of the Wii Remote and the actual was about $3\,[\mathrm{cm}]$.

The average cycle time of the simultaneous estimation is $43.2\,[\mathrm{ms}]$. This is nearly equal to the time of movement estimation with 40 features.

C. Results of Robot Control

In the experiment of robot control, the body movement and the arm motion of the user are estimated using the interface firstly, and then the commands for the robot are generated from the estimated temporal data and issued.

The robot, Enon, has a movement instruction with a 2D target position in its coordinate frame R (see Fig. 6), a head rotation instruction with target pan and tilt angles, and an arm motion instruction with a 3D target position. We set the target values as follows: the estimated horizontal position (x_I^W, z_I^W) is set to (y^R, x^R) of the movement instruction, the estimated pan and tilt angles of the orientation q_I^W is set to the head rotation instruction, and the estimated arm position p_A^I is set to the arm motion instruction, respectively. The estimated arm orientation q_A^I is not used in this experiment since the degrees of freedom of the robot arm is insufficient. The new command is issued when the previous command finished and the following conditions are fulfilled: the movement distance is larger than $0.5\,[\mathrm{m}]$ in the longitudinal direction and $1.0\,[\mathrm{m}]$ in the lateral direction for the movement (see Fig. 6), the sum of the pan and tilt moving angles is larger than $5\,[^\circ]$ for the head rotation, and the distance of the arm motion is larger than $0.05\,[\mathrm{m}]$ for the arm motion. The maximum speed of the robot movement is set to $0.4\,[\mathrm{m/s}]$.

Fig. 10 shows a result of robot control. Although there are some differences between the poses of the user and the robot, caused by the estimation error and the execution delay of the commands in the robot, the robot was able to follow the movement and the arm motion of the user.

VI. CONCLUSIONS

In this paper we have developed a wearable interface for controlling robot movement and arm motion based on measurement of human body motion. The wearable interface is composed of a camera, inertial sensors, and a Wii Remote. Using vision-inertial sensor fusion using EKF, drift-less and robust movement estimation is realized without any environmental sensors. The arm motion is estimated with the same framework as the movement estimation (sensor fusion using EKF and the same motion model). By eliminating the body movement from the arm motion, the interface can estimate the body movement and the arm motion simultaneously. In the robot control experiment, we succeeded in controlling the movement and the arm motion of the humanoid robot using the estimated body movement and arm motion.

The accuracy and the robustness of the current estimation are enough for just moving the humanoid as in our experiments. Further improvements are, however, needed for more complex robot motions such as grasping an object with a robot hand and bringing it to some specific position. In addition, feedback from the robot to the user such as visual or tactile feedback (*e.g.*, [10]) is also required for a safe robot control.

ACKNOWLEDGMENTS

This work is supported in part by Grant-in-Aid for Scientific Research (No. 23650096) from JSPS.

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*, pp. 397–402, 2001.

[3] T. Matsuyama, X. Wu, T. Takai, and S. Nobuhara, "Real-time 3d shape reconstruction, dynamic 3d mesh deformation, and high fidelity visualization for 3d video," *Computer Vision and Image Understanding*, vol. 96, no. 3, pp. 393–434, 2004.

[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*, pp. 4135–4140, 2007.

[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*, pp. 145–151, 2002.

[6] J. Sugiyama and J. Miura, "Development of a vision-based interface for instructing robot motion," in *Proc. of the 18th IEEE Int. Symp. on Robot and Human Interactive Communication*, pp.1198–1203, 2009.

[7] 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*, pp. 18–27, 2003.

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

[9] J. Montiel, J. Civera, and A. Davison, "Unified inverse depth parametrization for monocular slam," in *Proc. of Robotics: Science and Systems*, 2006.

[10] D. Tsetserukou, J. Sugiyama, and J. Miura, "Belt tactile interface for communication with mobile robot allowing intelligent obstacle detection," in *Proc. of IEEE World Haptics Conference*, pp. 113–118, 2011.

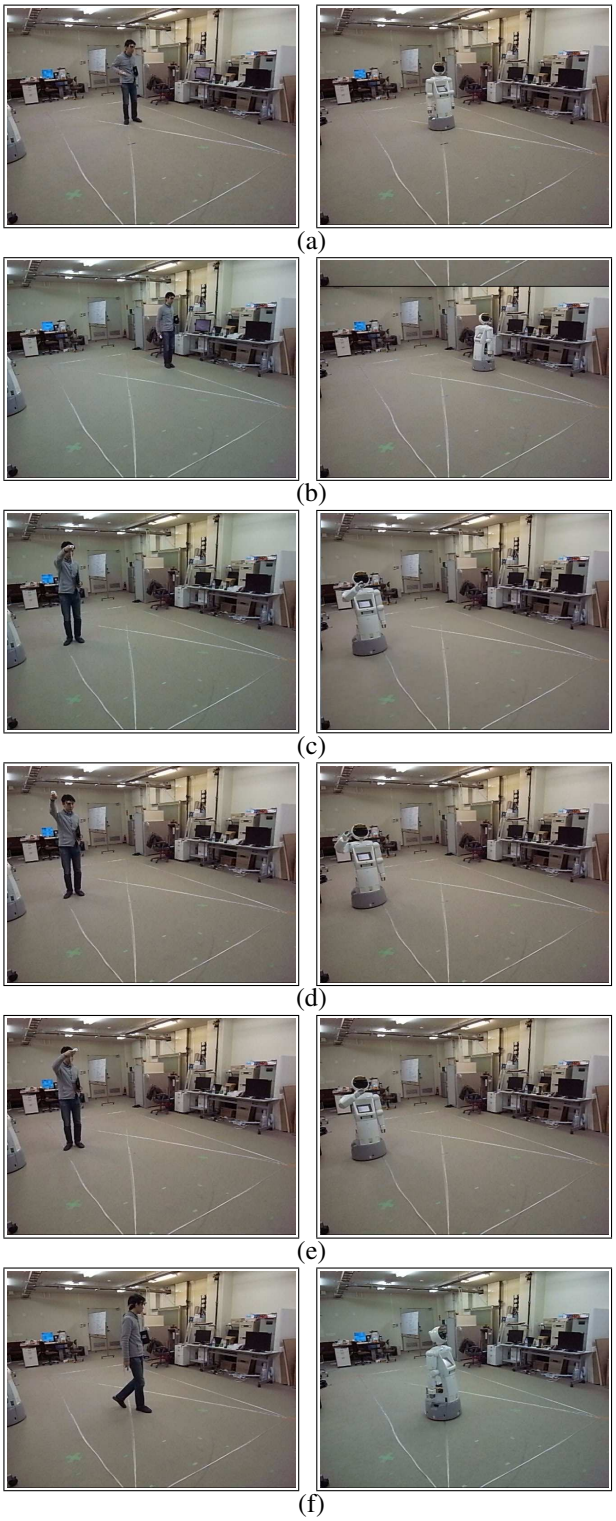


Fig. 10. Result of the robot control experiment. (a)–(b): the user moves leftward (0–5 [s] for the user, 0–9 [s] for the robot). (d)–(e): the user moves rightward and then moves the right arm (14–37 [s] for the user, 34–49 [s] for the robot). (f): the user moves to diagonally forward left (48 [s] for the user, 70 [s] for the robot).