Seamless Vision-assisted Placement Calibration for Wearable Inertial Sensors

JIAN WU and ROOZBEH JAFARI, Texas A&M University

Wearable inertial devices are being widely used in the applications of activity tracking, health care, and professional sports, and their usage is on a rapid rise. Signal processing algorithms for these devices are often designed to work with a known location of the wearable sensor on the body. However, in reality, the wearable sensor may be worn at different body locations due to the user's preference or unintentional misplacement. The calibration of the sensor location is important to ensure that the algorithms operate correctly. In this article, we propose an auto-calibration technique for determining the location of wearables on the body by fusing the 3-axis accelerometer data from the devices and three-dimensional camera (i.e., Kinect) information obtained from the environment. The automatic calibration is achieved by a cascade decision-tree-based classifier on top of the minimum least-squares errors obtained by solving Wahba's problem, operating on heterogeneous sensors. The core contribution of our work is that there is no extra burden on the user as a result of this technique. The calibration is done seamlessly, leveraging sensor fusion in an Internet-of-Things setting opportunistically when the user is present in front of an environmental camera performing arbitrary movements. Our approach is evaluated with two different types of movements: simple actions (e.g., sit-to-stand or picking up phone) and complicated tasks (e.g., cooking or playing basketball), yielding 100% and 82.56% recall for simple actions and for complicated tasks, respectively, in determining the correct location of sensors.

CCS Concepts: • **Computer systems organization → Sensors and actuators**

Additional Key Words and Phrases: On-body device localization, vision-assisted calibration, inertial measurement unit (IMU), Kinect, Wahba's problem

ACM Reference Format:
Jian Wu and Roozbeh Jafari. 2017. Seamless vision-assisted placement calibration for wearable inertial sensors. ACM Trans. Embed. Comput. Syst. 16, 3, Article 71 (July 2017), 22 pages.
DOI: http://dx.doi.org/10.1145/3023364

1. INTRODUCTION

Wearable computers are gaining significant popularity and are being widely used in various applications such as remote health [Hung et al. 2004, Jovanov et al. 2005], movement monitoring [Ghasemzadeh and Jafari 2011; Najafi et al. 2003], and human computer interface [Chen 2001; Kjeldskov and Graham 2003]. Of particular interest are inertial or motion-based wearable devices. Knowledge of the specific locations of

This work was supported in part by the National Science Foundation, under grants CNS-1150079 and ECCS-1509063, and the TerraSwarm, one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding organizations.
Authors' addresses: J. Wu and R. Jafari, 5045 Emerging Technologies Bldg./3120 TAMU, College Station, TX 77843-3120; emails: {jian.wu, rjafari}@tamu.edu.

DOI: http://dx.doi.org/10.1145/3023364

the wearable inertial devices on the body is a prerequisite for the development of most of the movement monitoring applications. Automatic calibration of the sensor locations is valuable due to three factors.

First, the auto-localization/calibration of the on-body devices will enhance the adaptability of these wearable devices. The design of pervasive wearable computers should enable the user to adjust the placement of the device according to their lifestyles, activities, and preferences. For example, a user prefers to put his cell phone in silent mode in his pocket if he is in a meeting, while he will prefer to keep it in his hand when he wants to make a phone call or browse the web. The user may also strap the cell phone to his/her arm as he/she is exercising to monitor his/her physical activities.

When it comes to the popular wearable activity trackers, some users may wear them on the wrist in the form of smart watches. Other users may wear them on their feet integrated into their shoes. Some users may prefer to wear them on the chest in the form of chest bands. The smart monitoring signal processing and applications should be aware of location changes and adaptively adjust themselves.

Second, the auto-localization of on-body sensors will make the collected data more reliable in cases where the sensor location information is missing or the incorrect information is entered by the user accidentally. With the emergence of the Internet-of-Things (IoT), trillions of sensors will be a part of daily human life [Lee et al. 2014]. It is very possible that the sensor configuration information that includes the on-body locations information will be missing for some sensors due to occasional errors. The auto-localization technique will help provide this information, thus making the data more useful and valuable.

Third, the auto-localization of the on-body sensors will reduce the setup time of highly dense and cooperative body sensor networks. One example is the motion sensors–based motion capture systems. They are attracting much attention due to their low cost, easy setup, and ability to provide pervasive sensing when compared to the classic marker-based motion capture systems using expensive cameras [Wu et al. 2014]. Although the setup time is significantly reduced compared with the marker-based system, which usually takes more than half an hour, it still requires a certain amount of effort to place each sensor to the assigned body segment [Weenk et al. 2013]. With the auto-localization techniques, the user just needs to place each of the sensors to different body segments, and the system will detect which sensor is assigned to which body segment. This feature can be leveraged as a service for wearable applications: the user puts the sensors on, and auto-calibration will determine on which body segments the sensors are placed. This information will then be relayed to the application to enable further signal processing.

Similarly to the wearable devices, the number of low-cost embedded sensing devices in the environment is increasing significantly. IoT infrastructures provide opportunities for wearable devices to interact with these devices in the environment [Roggen et al. 2009]. These opportunistic sensor fusion paradigms have the potential to enable many new applications, including the techniques under discussion in this article.

In this article, we propose a technique to determine the positions of the on-body wearable devices by fusing the wearable accelerometer data and the human skeleton information leveraged from 3D environmental cameras (i.e., Kinect). The two different modalities (i.e., accelerometer and Kinect) in the cyber world are coupled to observe the same physical entity (i.e., human body), thus enabling the calibration of one modality using the other one. The remainder of the article is organized as follows. The related work is reviewed in Section 2, and the preliminaries are introduced in Section 3. Our proposed calibration approach is explained in Section 4, followed by experimental setup in Section 5 and experimental results in Section 6. Finally, the conclusion is provided in Section 7.

2. RELATED WORKS

Several prior investigations have been proposed to localize the on-body locations of the wearable sensors. Most techniques attempt to recognize the walking activity as the first step, and after the walking is detected, the on-body sensor location is classified according to the training models. In one work, a sensor location and orientation-independent algorithm is used to detect walking, and then the specific motion characteristics of walking is leveraged to identify the on-body sensor locations [Kunze et al. 2005]. Their algorithms achieved a 100% accuracy for four sensor placements. Another proposed approach can obtain an average of 89% accuracy in estimating 10 different sensor placements with extensive experiments on 25 subjects [Vahdatpour et al. 2011]. In contrast to Kunze et al. [2005], the latter work uses an unsupervised technique to detect the walking activity such that the effort to define the models and patterns in the setup phase is not required, and the walking model is defined during runtime. However, due to the symmetry between the left arm and right arm or the right leg and left leg during walking, these algorithms may not be able to distinguish the location between the right leg and left leg or between the right arm and left arm. Another technique is proposed to recognize the locations of wearable sensors with a full body motion capture configuration (17 sensors) which was validated with 10 healthy subjects and 7 patients [Weenk et al. 2013]. This work calculates a global coordinate for each sensor using the 6s of walking at the beginning of each trial. The entire sensor data stream would be projected to this global frame such that the orientation information of the sensor is not required to detect the walking. A decision tree based on the C4.5 algorithm is developed to determine the sensor positions.

In all of the above works, the walking activity has to first be detected, and the on-body sensor localization is achieved by the classification algorithms specifically working with the walking activity. In reality, walking activities may not always be present. It will be more useful if the on-body location can be inferred from arbitrary activities. An approach to determine five different on-body sensor placements from arbitrary activities is proposed [Kunze and Lukowicz 2007]. This approach achieves up to 82% accuracy while it classifies the locations with a 6min window. However, there will be a large overhead for training the hidden Markov model (HMM), which needs to capture many arbitrary activities. In addition, the classification for 6min of data will be computationally expensive.

Our proposed approach calibrates the on-body sensor locations from the arbitrary activities by leveraging the information from an environmental camera and requires little to no training effort from the user at the setup phase. The work that comes closest to ours is described in Bahle et al. [2013]. In their work, the vertical angle change features are extracted for both five Kinect body segments and five inertial sensors, and dynamic time warping (DTW) [Berndt and Clifford 1994] is used to align the signal from two modalities to eliminate any time synchronization issues. Meanwhile, the Kinect segment that gives the smallest DTW distance is chosen for further consideration. The correlation between inertial and Kinect signals according to the time stamps is calculated, which serves as a confidence measure. The same procedure is repeated 5 times, and the body segment that offers the largest average confidence is determined as the final location. Unlike the work in Bahle et al. [2013], we formulate the problem as a 3D frame calibration problem, called Wahba's problem, and our method determines the on-body sensor localization by solving this problem. The solution to Wahba's problem will also calibrate the sensor frame to the Kinect frame which is an important step when these two modalities are used together for robust skeleton tracking applications [Helten et al. 2013; Pons-Moll et al. 2010]. Moreover, our approach only uses the accelerometers to recognize eight sensor locations instead of using the combination of accelerometers and gyroscopes to recognize five body locations in Bahle et al. [2013].

(a) Inertial sensor

(b) Kinect sensor

Fig. 1. Hardware description.

3. PRELIMINARY

3.1. Hardware Description

Figure 1(a) shows the 9-axis motion sensor with dimensions $1'' \times 1.5''$ that was designed and developed in our laboratory [Bennett et al. 2014]. An InvenSense MPU9150 9-axis microelectromechanical systems (MEMS) sensor is used to measure the 3-axis acceleration, 3-axis angular velocity, and 3-axis magnetic strength. A Texas Instrument 16-bit low-power microcontroller (MSP430F5528) is used as the central processor. Both a dual mode Bluetooth module and a microSD card unit are available on the board. The user can choose to stream the data to a PC/tablet for real-time processing or log all the data to the microSD card for long-term movement monitoring. A charging circuit is included for the battery. Kinect is a low-cost RGB-Depth motion sensing device developed by Microsoft as shown in Figure 1(b). It is widely used in applications of motion tracking [Oikonomidis et al. 2011, 2012] and rehabilitation [Chang et al. 2011; Lange et al. 2011]. Microsoft provides an application program interface (API) to obtain the joint positions of the human body as captured by the Kinect, enabling real-time skeleton tracking. In this article, the 3D joint positions and joint tracking states are used in our algorithm. The body segment vectors are constructed from positions of every two adjacent joints.

The accelerometer is the principal sensing modality used in this article. The gyroscope and magnetometer sensors are not used in this article. The magnetometer measures the magnetic field strength and is used as e-compass in some mobile phone applications. However, it suffers from severe magnetic interference from environment (e.g., the existence of metal pipes) and requires extensive calibration for different locations [Crassidis et al. 2005]. The gyroscope measures 3-axis angular velocity and is usually used with accelerometer as inertial navigation system (INS). However, the power consumption of the gyroscope is much higher than accelerometer (e.g., the power consumption of gyroscope is more than 10 times the accelerometer in the MPU9150 that we are using). As power consumption is crucial to a wearable device,

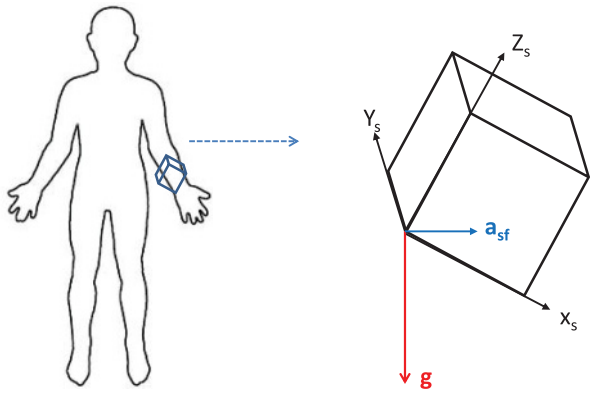

Fig. 2. Example of accelerometer measurement during horizontal arm lifting movement.

many commercial wearable devices are only equipped with accelerometers for activity recognition (e.g., Adidas Fit Smart, Fitbit Flex, Jawbone Up Move, Sony SmartBand Talk). The power consideration is the main reason only an accelerometer is used in our approach. Recently, very few wearable devices have begun to incorporate a gyroscope to provide better motion tracking ability, although the battery lifetime is adversely impacted. We will look into adding a gyroscope in our approach in future work. The 3-axis accelerometer measures the gravitational acceleration and non-gravitational acceleration caused by motions (known as force specific acceleration). The force-specific acceleration is also called dynamic acceleration in our article. Figure 2 shows an example of how gravitational acceleration and force specific acceleration are measured by an accelerometer during an arm-lifting movement of the user. The accelerometer measures acceleration a_x, a_y, and a_z in its local frame *Xs*, *Ys,* and *Zs*, where

$$a_x = a_{xg} + a_{xsf}, \tag{1}$$

$$a_y = a_{yg} + a_{ysf}, \tag{2}$$

$$a_z = a_{zg} + a_{zsf}. \tag{3}$$

$a_{xg,}$ a_{yg}, and a_{zg} are decomposed values of gravitational vector g along sensor local frame *Xs*, *Ys*, and *Zs*. a_{xsf}, a_{ysf}, and a_{zsf} are decomposed values of specific force vector a_{sf} along the *X-axis*, *Y-axis,* and *Z-axis* of the sensor local frame. The gravitational acceleration g is caused by Earth's gravity and points to the surface of the Earth. The specific force is caused by motion and, in this example, it is caused by a horizontal arm lifting movement. More details are presented in Groves [2013] and Savage [1998].

3.2. Definitions

Before introducing the details of our approach, the term definitions are summarized in Table I to enhance the readability of the article and equations. We refer to each sample or data point as a frame. For example, if the camera is capturing video at 30 frames per second, we will have 300 frames over 10s. If the accelerometer is sampling at 200Hz, then we will have 400 frames over 2s. The 3D accelerometer measures acceleration, which is a combination of acceleration observed due to movements and the gravity along three axes of the sensor, and at each frame or sample it generates a 3×1 vector. f_{aj}^{i} denotes the ith frame of jth accelerometer data. j is the index for each wearable accelerometer and we are considering determining the location for multiple accelerometers. The Kinect API captures the human skeleton, from which all joints

Table I. Term Definition

Term	Definition
f^i_{Km}	ith frame of mth Kinect body segment vector, it is a 3×1 vector
f^i_{aj}	ith frame of jth accelerometer data vector, it is a 3×1 vector
$d*$	the asynchronization delay between accelerometer data stream and Kinect data stream
n	Total number of samples in a window
$\boldsymbol{a_{ji}}$	ith normalized jth accelerometer data vector in a window of n samples, it is a 3×1 vector
$\boldsymbol{k_{mi}}$	ith normalized mth Kinect body segment vector in a window of n samples, it is a 3×1 vector
$\boldsymbol{A_{jm}}$	Rotation matrix that transforms jth accelerometer vector to mth Kinect body segment vector, 3×3 matrix
$E(\boldsymbol{A_{jm}})$	The least-squares error after transforming all n samples of jth accelerometer vectors to n samples of mth Kinect body segment vectors in a window.
e_{jm}	The least-squares error between jth accelerometer and mth body segment after solving wahba's problem

positions in the Kinect frame are obtained. Two joints of a body segment construct a body segment vector. At each frame, Kinect captures all joint positions, and segment vectors are constructed. Each segment vector is a 3×1 vector. f^i_{Km} is the ith frame of the mth Kinect body segment vector. m is the index of each body segment (e.g., arm, thigh). If the accelerometer sensor and Kinect sensor are perfectly synchronized, then f^i_{aj} and f^i_{Km} measure movements occurring at exactly the same time. However, it is not guaranteed that they will be strictly synchronized to each other and a delay of d^* frames may exist between two streams. This is because each sensor modality may have its own clock. In IoT settings, we can always expect a lack of perfect synchronization and delays due to wireless communications. Our formulation handles this issue. In this article, we consider windows of data, and n is the number of samples in one window. $\boldsymbol{a_{ji}}$ is the normalized vector of f^i_{aj}, and $\boldsymbol{k_{mi}}$ is the normalized vector of f^i_{Km}. $\boldsymbol{A_{jm}}$ is a rotation matrix that transforms jth accelerometer vectors to mth Kinect segment vectors in a window. e_{jm} is the least-squares error between the jth accelerometer and mth body segment after solving Wahba's problem.

4. PROPOSED APPROACH

In this investigation, we use eight wearable sensors attached to eight body segments, which are listed in Table III. The body segments can be captured by Kinect cameras. The objective of our approach is to find the correct location information for all 8 accelerometers. To achieve this, we use observations made by two modalities and attempt to match them with each other. The accelerometer measures both dynamic acceleration due to movement and the gravity vector. The Kinect body segment vector measures the same vector change as gravity change in the Kinect frame. Although the Kinect body segment does not measure the gravity directly, the rotation of the body segment in the Kinect frame is the same as the rotation of the gravity vector in sensor frame. Thus, the accelerometer and Kinect segment observe the same rotations from two different frames. Since the accelerometer measures the dynamic motion at the same time, we applied a weighting method to reduce the impact of this part of acceleration, which is explained in Section 4.3.3. Our approach transforms accelerometer-measured gravity vector observations in a window to Kinect measured segment observations. If a sensor

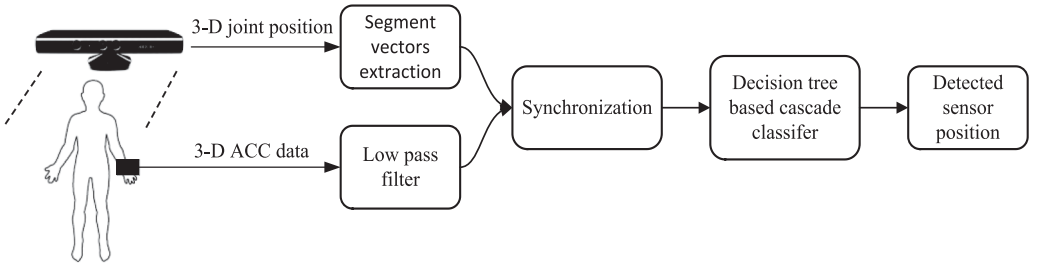

Fig. 3. Diagram of proposed approach.

is attached to a segment, after transforming the accelerometer vector observations to the Kinect frame, then they will become the same vector observations as this Kinect body segment vector observations. Therefore, we will get smallest transformation error for this pair of accelerometer and Kinect segment compared to errors obtained between this accelerometer and other Kinect segment vectors. This will be explained in detail in Section 4.2.

This is the core idea of our approach. Our proposed approach is shown in Figure 3. Motion sensors measure the 3D acceleration (ACC) data in the sensor local frame, and the Kinect measures 3D joint positions of the human skeleton in the Kinect frame. A low-pass filter is applied to the ACC data to remove the high-frequency noise. Details of the preprocessing are articulated in Section 4.1. The joint position data are used to construct the body segment vectors. We formulate the problem as the Wahba's problem and solve for the frame differences between the ACC vector observations measured from one accelerometer and all the segment vector observations from eight body segments measured in the Kinect frame. The exact body segment that has the sensor will observe the same movement and thus the accelerometer vector observations and this body segment vector observations are exactly same vector observations in two different frames. For the accelerometer vector observations and the other body segment vectors that do not include this sensor, if the accelerometer vector observations are transformed to the other body segments vector observations, there will be a larger error. The errors will be described in detail in Section 4.2. Using a minimum least-squares error acquired after solving the Wahba's problem between an accelerometer and all body segments, the sensor is assigned to a certain body segment by a decision-tree-based cascade classifier. The details are presented in Section 4.3.

4.1. Preprocessing

The accelerometer will have high-frequency noise that generates a large amplitude peak in the signal. Since we are calibrating the coordinate differences frame by frame, if the accelerometer vector with the high-frequency noise is used, the calibration error will increase. As most human activities of daily living are at a low frequency and to remove the high-frequency noise that exists in accelerometer data, a low-pass filter is usually applied with a cut-off frequency of 4Hz–8Hz [Bartlett 2007]. In our article, 5Hz low-pass filter is applied. From the Kinect skeleton data, the joint positions are obtained. Since the body segment vector information is required, we construct the segment vectors from joint positions.

Synchronization of the two systems is important to our approach, because our algorithm assumes each pair of vectors in the two different coordinate frames is observed at the same time. However, our approach adapts the idea of opportunistic sensing and the wearable accelerometer and vision sensor are fused in an opportunistic manner. For example, a user wears his music player, which has an accelerometer, on his arm.

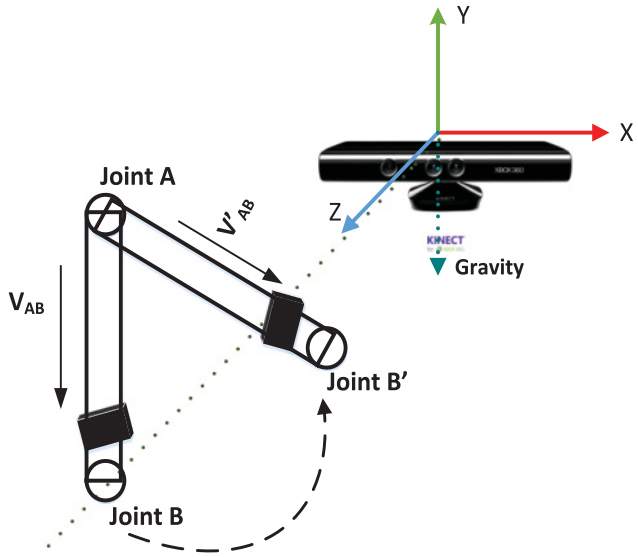

Fig. 4. Example of a body segment rotation.

When he enters the gym and performs exercises, the camera at the door captures his skeleton information. The skeleton information and accelerometer data are both sent to the cloud, and our algorithm calibrates the location sensor on the cloud. Because of the opportunistic nature of our approach, the accelerometer and camera are possibly not well synchronized. The sampling rate of the motion sensor is 200Hz and the sampling rate of the Kinect is 30Hz. The Kinect samples are interpolated using cubic splines to 200Hz. Canonical correlation analysis has proved to be effective to synchronize the audio and video features before fusing them [Sargin et al. 2007]. In this article, we adopt the same method to synchronize the Kinect and accelerometer data. As discussed in Section 3.2, we denote the mth Kinect body segment and jth accelerometer data of the ith frame by f^i_{Km} and f^i_{aj}, respectively. The problem becomes finding the delay d^* to maximize the mutual information between the Kinect and accelerometer. After d^* is obtained, the two modalities are synchronized by shifting d^* samples.

4.2. Wahba's Problem Formulation

Wahba's problem, first posed by Grace Wahba in 1965, seeks to find a rotation matrix between two coordinate systems from a set of (weighted) vector observations [Wahba 1965]. In our technique, the accelerometer and Kinect segment observe the same physical movement if the accelerometer is attached to this body segment. It means that 3D accelerometer observations and 3D body segment observations from Kinect are the same observations from two different frames. This is explained using Figure 4. Two joints, A and B, construct a body segment. Kinect measures 3D position data of joints A and B and the vector $\mathbf{V_{AB}}$ can be easily constructed. Assume that this body segment is placed in parallel to gravity; $\mathbf{V_{AB}}$ can be thought of in the same way (a vector) as gravity, since both of them will be normalized to unit vectors in our later formulation. As a body segment moves from AB to AB′, the body segment vectors measured by Kinect during this period can be considered as the gravity vector rotates from the AB to AB′ position. It is known that the gravitational acceleration measures the rotation of gravity in the Earth frame [Madgwick 2010]. Thus, the normalized vectors measured by Kinect and the normalized gravitational vectors measured by the accelerometer can

be considered as the same vectors measured in different reference frames. As we discussed in Section 3.1, the accelerometer measures not only gravitational acceleration but also specific force acceleration. Since there is no easy way to separate them, the specific force acceleration will be considered as noise in our algorithm.

Let $\boldsymbol{a_{ji}}$ be the ith normalized vector of the jth accelerometer in a window of n observations. $\boldsymbol{k_{mi}}$ is the corresponding normalized mth body segment vector measured in the Kinect frame. The normalization procedure for the accelerometer observation vector and the Kinect body segment vector are the same. Both of them are three-dimensional (3D) vectors and are normalized to corresponding unit vectors, which are required by the Wahba problem formulation. $\boldsymbol{A_{jm}}$ is a 3×3 rotation matrix that transforms the jth accelerometer vectors to mth Kinect body segment vectors. The rotation matrix is an orthogonal matrix. To solve for the rotation matrix, we attempt to minimize the least-squares error cost function as shown in Equation (4). In Equation (4), $E(\boldsymbol{A_{jm}})$ is the weighted least-squares error between all n observations of mth Kinect body segment and n transformed jth accelerometer observations with rotation matrix $\boldsymbol{A_{jm}}$. Equations (5)–(10) show how the rotation matrix $\boldsymbol{A_{jm}}$ is determined and after $\boldsymbol{A_{jm}}$ is identified; we will obtain the least-squares error $E(\boldsymbol{A_{jm}})$. This error is the principal deciding parameter used with our classifier at a later stage,

$$E(\boldsymbol{A_{jm}}) \equiv \frac{1}{2} \sum_{i=1}^{n} w_i |\boldsymbol{k_{mi}} - \boldsymbol{A_{jm}} \boldsymbol{a_{ji}}|^2, \tag{4}$$

where w_i is non-negative observation weight. The cost function can be rewritten as

$$E(\boldsymbol{A_{jm}}) = \sum_{i=1}^{n} w_i - \sum_{i=1}^{n} w_i \boldsymbol{k_{mi}^T} \boldsymbol{A} \boldsymbol{a_{ji}}. \tag{5}$$

Next, we rewrite the cost function as

$$E(\boldsymbol{A_{jm}}) = \lambda_0 - tr\left(\boldsymbol{A_{jm}} \boldsymbol{B_{jm}^T}\right) \tag{6}$$

with

$$\lambda_0 = \sum_{i=1}^{n} w_i. \tag{7}$$

and where $\mathbf{B_{jm}}$ is defined as

$$\boldsymbol{B_{jm}} \equiv \sum_{i=1}^{n} w_i \boldsymbol{k_{mi}} \boldsymbol{a_{ji}^T}, \tag{8}$$

where tr is the trace of a matrix. The first useful solution to Equation (6) was provided by Paul Davenport in Lerner [1978]. The same approach is used in this article to solve this problem and Equation (6) can be rewritten with the quaternion representation as

$$E(\boldsymbol{A_{jm}}(\boldsymbol{q})) = \lambda_0 - \boldsymbol{q^T} \boldsymbol{K}(\boldsymbol{B_{jm}}) \boldsymbol{q}, \tag{9}$$

where $K(\mathbf{B_{jm}})$ is the symmetric traceless 4×4 matrix as shown in Equation (10) and $\boldsymbol{q}$ is a 4D quaternion that represents the rotation transformation instead of rotation matrix $\mathbf{A_{jm}}$,

$$K(\boldsymbol{B_{jm}}) = \begin{bmatrix} \boldsymbol{B_{jm}} + \boldsymbol{B_{jm}^T} - I_{3\times3} tr(\boldsymbol{B_{jm}}) & \sum\limits_i w_i \boldsymbol{k_{mi}} \times \boldsymbol{a_{ji}} \\ (\sum\limits_i w_i \boldsymbol{k_{mi}} \times \boldsymbol{a_{ji}})^T & tr(\boldsymbol{B_{jm}}) \end{bmatrix}. \tag{10}$$

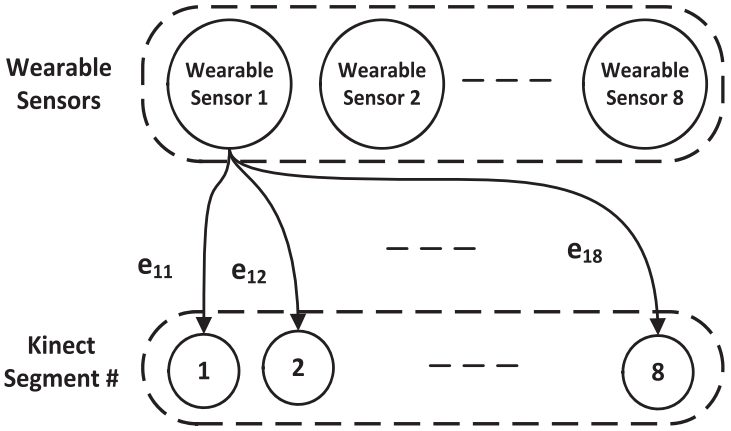

Fig. 5. The least-squares errors between eight wearable sensors and eight Kinect body segments.

It follows that the optimal quaternion is the eigenvector of $K(\mathbf{B_{jm}})$ with the maximum eigenvalue:

$$K(\boldsymbol{B_{jm}})\boldsymbol{q_{opt}} = \lambda_{max}\boldsymbol{q_{opt}}, \tag{11}$$

where $\boldsymbol{q_{opt}}$ is the optimal quaternion solution to Equation (9) and λ_{max} is the maximum eigenvalue of matrix $K(\mathbf{B_{jm}})$.

After the optimal quaternion is obtained, the least-squares mean error when calibrating the two frames is determined. In this article, we assign eight wearable accelerometers to eight Kinect segments. The eight Kinect segments we consider are the right upper arm, right forearm, left upper arm, left forearm, right thigh, right lower leg, left thigh, and left lower leg. These are the eight possible body locations where a given sensor will be worn. The eight errors between one sensor and eight body segments are used to determine the location of this sensor in the following section. Although we use the above-mentioned eight body segments for our experimental validation, our technique can operate with a larger number of body segments.

4.3. Auto-localization

Figure 5 shows the least-squares errors between the wearable sensors and the Kinect body segments. The nodes in the top show eight wearable nodes that need to be localized. The nodes in the bottom part of the figure represent eight Kinect segments that are discussed in this article. e_{jm} is the least-squares error between the jth wearable sensor and the mth Kinect segment solved in a window of observation as discussed in Section 4.2. The Kinect segments are shown in Table III of Section 5. For each wearable sensor, we have eight errors. The eight errors are the inputs for one cascade classification to determine the location of this sensor. If there are n wearable sensors on the human body, then they are treated independently, and the same cascade classifier will be used for all of them. In this article, eight independent cascade classifiers are needed to determine the locations of eight wearable sensors. In the following section, the operation of the classifier for one wearable sensor is discussed.

4.3.1. Cascade Classifier. To determine the location of an on-body sensor, a cascade classifier is proposed, as shown in Figure 6. In our approach, multiple windows are used to determine the location of the sensor. Among all windows, some windows offer well-pronounced motions and some other windows may not contain motion, which will reduce their suitability to aid in the localization of on-body sensors. Thus, several

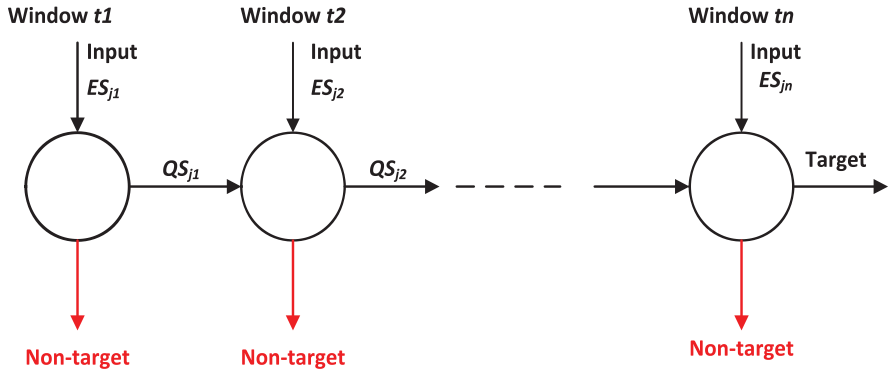

Fig. 6. Cascade classifier with multiple decision nodes for the jth wearable accelerometer. Each cycle represents a decision-tree-based classifier at each time window. The decision-tree-based classifier is explained in Section 4.3.2. Each classifier has its own input error set ES. ES_{jl} is the input error set for the lth classifier of the jth sensor. The output of each classifier is called qualifying set (QS), which may include multiple body segments that can be assigned to the jth wearable accelerometer. The cascade will output the QS_{jn} as the target movement when the size of QS_{jn} becomes 1 and all the non-target locations are rejected. An example is given in Section 4.3.1.

windows must be considered over time and the notion of a cascade classifier is used. A cascade classifier is widely used in object tracking and face recognition [Lienhart et al. 2003; Viola and Jones 2004]. The cascade classifier has a series of decision nodes, since one single decision node is not enough to determine the classification result. Each node will reject some non-target classification labels and the remaining potential candidates will be passed to the next node. In our case, a single decision node is not able to generate a single sensor location, as it can reject only some non-target locations. Therefore, a cascade classifier is ideal for our application, and each decision node is a decision-tree-based classifier, as discussed in Section 4.3.2. The input of the lth decision node is error set ES_{jl}, which is defined to include the least-squares errors between the jth accelerometer and all the remaining qualified Kinect segments. The qualified Kinect segments are the ones that have not been rejected yet. The output will be the remaining qualified Kinect segment indexes after the non-target locations are rejected by this node. These indexes form the qualifying set for each node. The qualifying set for the lth decision node for the jth wearable sensor is defined as QS_{jl}. The following example shows how the cascade classifier works. To determine the location of the jth accelerometer, the input of the first decision node ES_{j1} is $\{e_{j1}, e_{j2}, \ldots, e_{j8}\}$, since all the locations are candidate locations. Based on the decision-tree-based classifier, which is discussed in Section 4.3.2, if the locations 1 and 2 are rejected by this decision node, then the output qualifying set QS_{j1} is $\{3, 4, \ldots, 8\}$. In the second decision node, the input ES_{j2} is $\{e_{j3}, e_{j4}, \ldots, e_{j8}\}$, in which we only consider the qualifying body segments. Similarly, more decision nodes are cascaded until QS_{jn} only has one element, which means seven non-target locations are rejected by the former nodes and the only one remaining segment will be the target location of jth wearable accelerometer. For each decision node, the input error between accelerometer samples and corresponding Kinect segments samples is considered over a 1.5s window. The window size is chosen as 1.5s, since it is the time duration for most of the daily activities (e.g., sit-to-stand and pick up a cup). The overlapping segment between two consecutive windows is set at 0.5s.

4.3.2. Decision-Tree Classifier. Each decision node in the cascade classifier itself is a decision-tree-based classifier that will reject some non-target locations. Figure 7 shows the decision-tree classifier, which is the classifier for each decision node in Section 4.3.1.

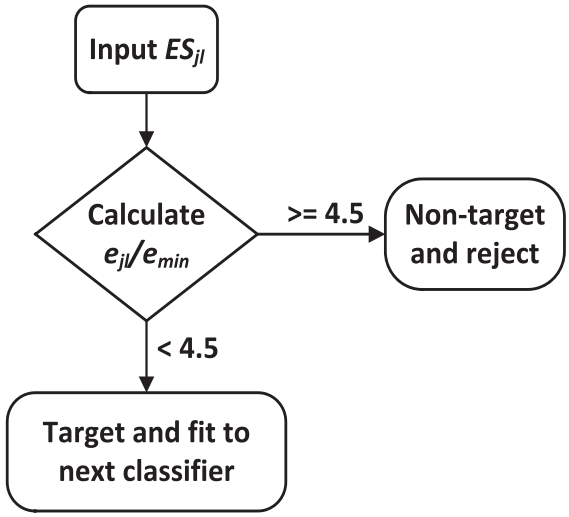

Fig. 7. Decision-tree classifier.

The input of the lth decision node is the error set ES_{jl}. The minimum error from the error set is e_{min}, and for each error e_{jl} in this set, we calculate e_{jl}/e_{min}. We consider this ratio instead of using the minimum error, because when two or more Kinect body segments experience no movements, they will report similar errors. For example, if a user remains in a sitting position and his lower body does not report significant motion, with a sensor attached to the right thigh, then all lower body segments, including the right thigh, right lower leg, left thigh, and left lower leg will have similar errors, and the smallest is not necessarily the right thigh. The error should be sufficiently discriminative to assist the classifier choosing the correct body segment. If the result is smaller than a threshold, then the corresponding location will be classified as a possible target location, and the index will be provided to the next classifier. If the result is larger than or equal to the threshold, then the corresponding segment will be classified as non-target location and will be rejected. The threshold is set at 4.5, which is determined experimentally.

4.3.3. Weighting Method. From the problem formulation in Equation (4), there is a weight parameter w_i for each frame of error calculation. If all the observation samples in one classifier window are of equal importance, then all the w_i will have equal values in this window, and all the values need to be normalized in this window. However, the confidence of the accelerometer observations is related to the speed of the movement. It is well known that the accelerometer measures the combination of the gravitational and dynamic acceleration [Mizell 2003], and the gravitational acceleration is useful information for our algorithm to track the rotations of body segments. If the dynamic acceleration is too large, then our estimation error will increase. The raw weight for observation sample i is defined as

$$w_i = \begin{cases} 0, ||f_{aj}^i| - g| > 0.5g \\ g^2/(g + ||f_{aj}^i|_i - g|)^2, ||f_{aj}^i| - g| \leq 0.5g \end{cases}, \tag{12}$$

where g is the gravity constant. In Equation (12), if the absolute value of difference between total acceleration amplitude and gravity constant is larger than 0.5g, the weight is set to 0. It means if the amplitude of dynamic acceleration is larger than 0.5g,

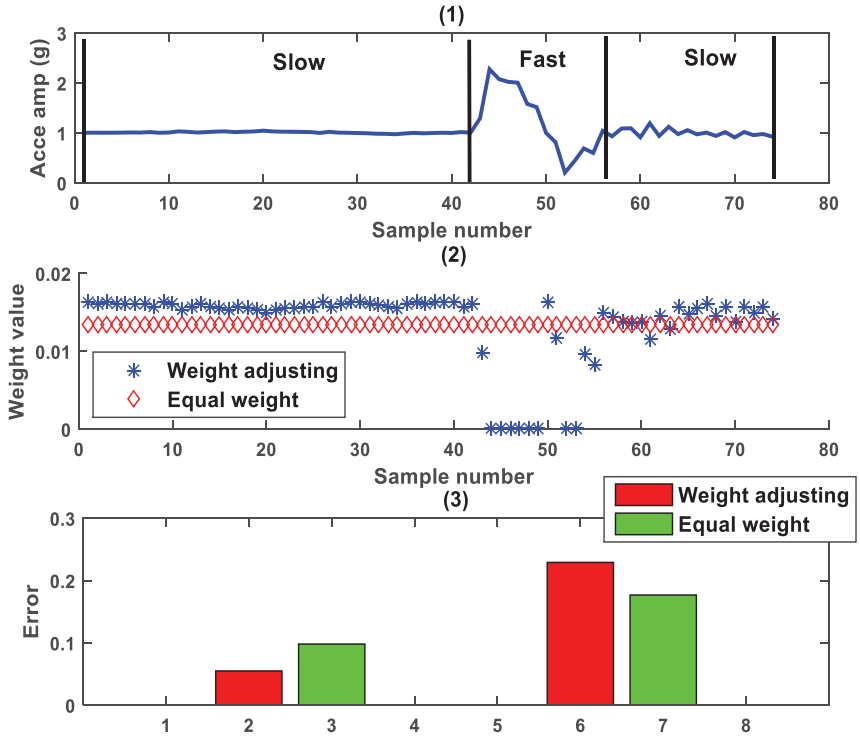

Fig. 8. (1) Acceleration amplitude of an arm stretch activity, including fast and slow motion. (2) Normalized equal weights and adjusted weights. (3) Errors for target location and average of non-target locations of using a weight adjusting approach and an equal weight approach.

then the impact of dynamic acceleration is considered too high, and we do not consider this sample. When the absolute value of the difference is smaller than 0.5g, we use the sample but its weight is penalized based on how much it differs from the gravitational acceleration g. Normalizing the raw weight in this window, we will obtain the weights as

$$w_i' = w_i \Big/ \sum_{i=1}^{n} w_i. \tag{13}$$

Figure 8 illustrates how weight adjusting method performs better than if equal weights are assigned to all samples for an arm stretch movement with combined fast and slow motion. The user is asked to perform the complete movement at a slow speed at first and then at a fast speed, followed by another slow-speed movement. The sensor is attached to the right forearm. Figure 8(1) shows the acceleration amplitude during this activity, and Figure 8(2) shows our proposed weight adjusting method, which gives the slow-motion higher weight and the fast-motion smaller weight, since the gravity vectors are affected more by the dynamic motion acceleration during the fast motion. Figure 8(3) illustrates how weight adjusting method achieves smaller error between sensor measurement and the target Kinect segment and a larger average error between the sensor and non-target segments. This will in turn yield better separation between the target and non-target locations.

As stated, we are using the rotations of gravity on the wearable accelerometer and the Kinect body segment to determine the location of the sensors. The gyroscope can also

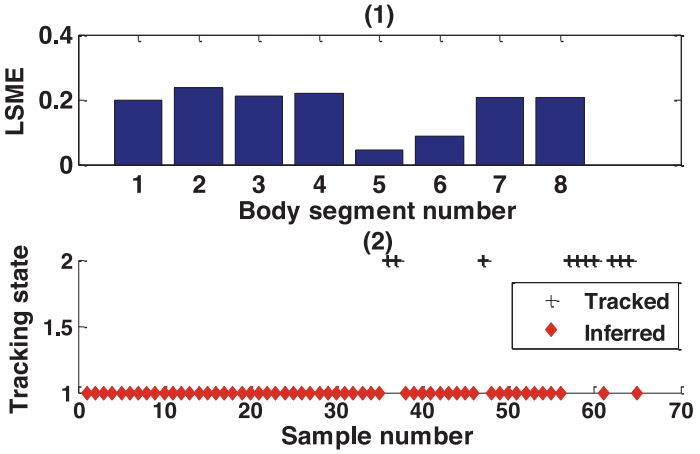

Fig. 9. (1) Least-mean-squares errors between the left lower leg sensor and eight Kinect body segments. (2) Tracking state of the left foot joint during the left lower leg kneeling activity.

give us the same information. There are three reasons why we do not use gyroscopes. First, we will need to perform integration over gyroscope readings to obtain rotation information that will introduce integration error impacting our localization algorithm. Second, the accelerometer consumes less power, and it is readily available in most wearable sensors. Third, the gyroscope captures the rotation along the vertical direction of the Earth while Kinect segment vectors do not offer this information. This difference will lead to poor performance if a match is done between these two modalities.

4.3.4. Kinect Occlusion Consideration. Kinect suffers from the occlusion (i.e., line of sight) problem when some parts of the body are not visible. If an occlusion for a joint occurs, then the position data will not be correct, which will decrease the performance of our algorithm. To solve this problem, we look at the skeleton joints tracking state from the Kinect API. The API offers three states: "tracked," "inferred," and "non-tracked." When the tracking state is "inferred," it means the data are calculated from the other tracked joints and the confidence in the data is low. The state "non-tracked" means the joints are not available. In this article, if there are any "inferred" and "non-tracked" state for any joint during the decision window, this decision window will not be considered and the algorithm will process the next window. The reason for such a strict constraint is that if the tracking status of one joint is not reliable, then it may result in a wrong body segment vector that consists of this joint. The wrong body segment vector may result in a false positive that adversely affects the decision of our algorithm. In future work, we will improve our algorithm to also operate with "inferred" states.

Figure 9(2) shows that during the kneeling movement with the left lower leg, the left foot joint is inferred for some samples, and the least-squares mean errors between the left lower leg sensor and the eight Kinect segments are shown in Figure 9(1). Because of the effect of the occlusion of the left foot, the error between the left lower leg sensor and the left lower leg (#8) is much larger than the error with respect to the non-target segment right thigh (#5). This will possibly result in misclassification, and hence it is very important to eliminate these illegal cases before the classification takes place.

5. EXPERIMENT SETUP

To test the effectiveness of our method, two experiments are designed. In the first experiment, we test our algorithm with a number of individual activities, which take about 1.5s to complete. The activities are listed in Table II. For most activities, we list two labels: left and right. These labels refer to which side of the body is used to perform

Table II. Daily Activity List

No(#)	Activity	No(#)	Activity
1	Bend and grasp (right)	7	Sit to stand
2	Bend and grasp (left)	8	Stand to sit
3	Kick (right)	9	Arm lateral raise (left)
4	Kick (left)	10	Arm lateral raise (right)
5	Leg lifting (right)	11	Pick up phone (right)
6	Leg lifting (left)	12	Pick up phone (left)

Table III. Sensor Placement List

No(#)	Sensor location	No(#)	Sensor location
1	Right upper arm	5	Right thigh
2	Right forearm	6	Right lower leg
3	Left upper arm	7	Left thigh
4	Left forearm	8	Left lower leg

the activity. For example, "Bend and grasp (right)" means this activity is performed by right arm. In the second experiment, two complicated daily tasks—cooking and playing basketball—are considered. For the first experiment, five subjects perform 10 repetitions for each activity. For the second experiment, the same subjects perform 3min of each task 3 times in the experiment area. Table III shows the eight sensor placements for our experiment and their corresponding index numbers.

6. EXPERIMENT RESULTS

6.1. Results of Simple Activities

Note that the durations of the activities we investigate are about 1.5s and only one window is used to determine the location. Thus, the outputs will be the outputs of our first stage classifier. To look at the discriminative ability of each activity for each location, two classification performance metrics (precision and recall) are discussed. The definitions of precision and recall are as follows [Powers 2011]:

$$precision = \frac{tp}{tp + fp}, \tag{14}$$

$$recall = \frac{tp}{tp + fn}, \tag{15}$$

where tp is the number of true positives, fp is the number of false positives, and fn is the number of false negatives for location recognition. We have eight locations with one sensor on each location and the objective is to find a matching between sensors and locations. The performance measures are considered for each location. If a sensor is attached at one location, and it is classified to this location, then it would be considered as tp. If the sensor is attached to another location and it is assigned to this location, then it would be considered fp for this location. If this sensor is classified as another location instead of the correct location, then it would be considered as fn. For each location, the precision is the ratio between correctly recognized instances for this location and the summation of correctly recognized instances and the instances that are recognized as this location incorrectly by the algorithm. This measure provides how well the algorithm can separate the false positives that are confused with the correct location. The recall is the ratio between number of correctly recognized instances for one location and the summation of the number of correctly recognized instances and the number of instances that belong to this location and recognized as other locations incorrectly. It measures how well the algorithm recognizes the correct sensor locations.

Figure 10 shows the recall and precision for all the locations from the 12 different activities. The values are averaged for five subjects. From the three plots, we can see

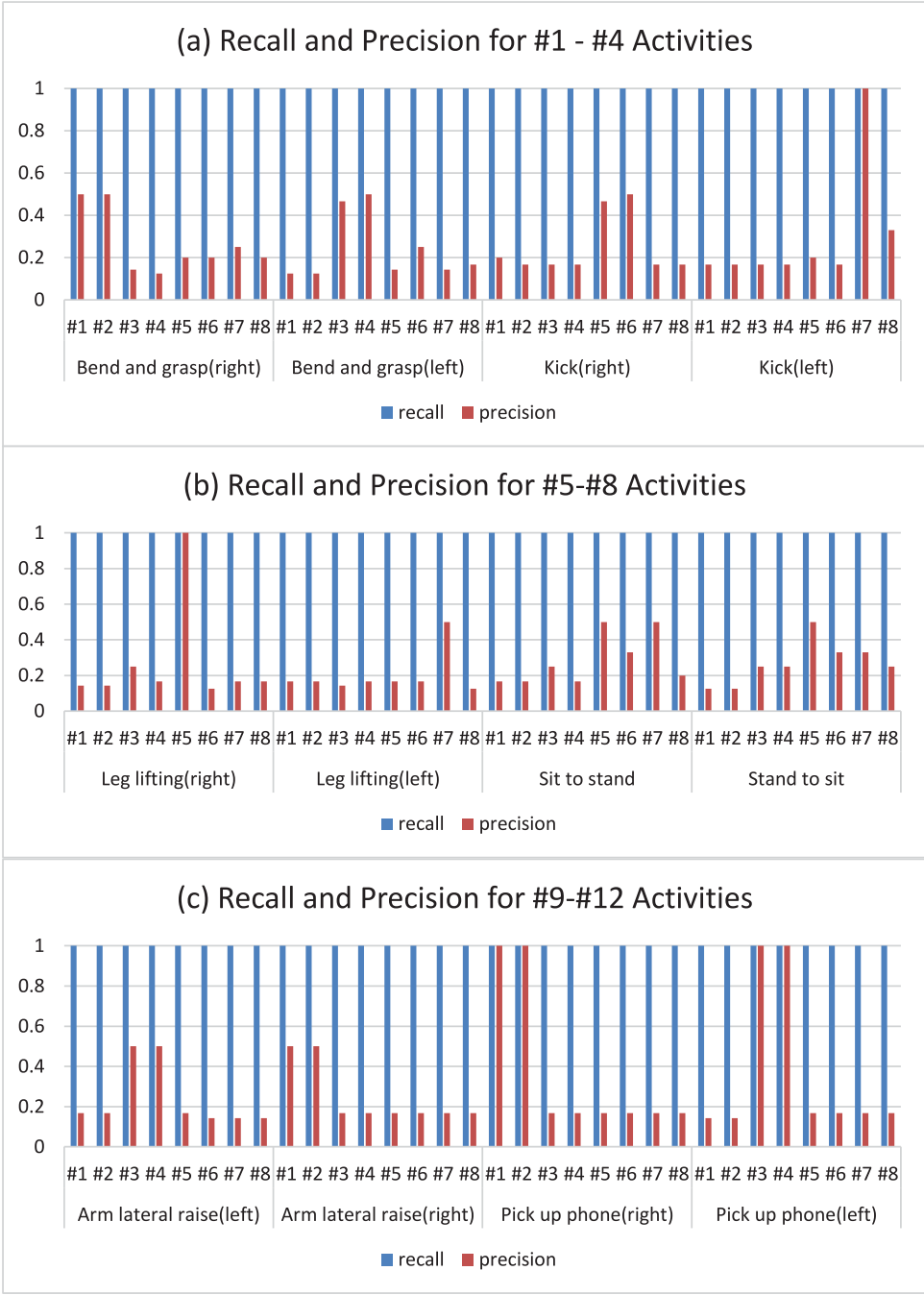

Fig. 10. Recall and precision for eight locations from (a) #1–#4 activities, (b) #5–#8 activities, and (c) #9–#12 activities.

that the recalls for all locations from all the activities are 100%, which means our algorithm will not reject the true positives for the 12 daily activities. Note that for all the simple activities, we can only investigate one-stage classification as the movements are short (1.5s) and a high recall rate is the most important metric, since we do not want the true positive instances to be rejected at every stage. In Figure 10(a), for the Bend and grasp (right), the precisions of right upper arm (#1) and right forearm (#2) are both at 50%, which means only one additional false positive is detected. For this movement, the right upper arm and right forearm cause false positives for each other, since they have similar motions. The same result is observed for bend and grasp (left). For the first two movements, other locations have a smaller precision, because when the sensor is attached to other locations, those body segments do not observe movements leading to potential confusion among these body segments. For the kick (right) and kick (left) activities in Figure 10(a), only thighs (#5 or #7) and lower legs (#6 or #8) of the corresponding body side (right and left) have larger precision and all other locations have a similar small precision, since they all do not move during the activities. For leg lifting (right) and left lifting (left) in Figure 10(b), the right thigh (#5) and left thigh (#7) have large precision values, respectively. It means leg lifting will reject most of the false-positive cases when the sensor is attached to the thighs. In these two activities, even though lower legs (#6 and #8) have major movement, they have small precision values. This is because in the leg lifting movement, the lower legs are almost parallel to gravity. They do not have major change along the gravity, and thus the gravitational change measured by accelerometer is very small. Since the gravitational acceleration change is the useful information to our algorithm and the force specific acceleration is considered noise, low precision values are achieved. For the sit to stand and stand to sit, the precisions of both thighs (#5 and #7) are bigger than the others due to the major motion occurring at the thighs and our algorithm will give much smaller least-squares errors between motion sensor and thighs than between motion sensor and the other body segments. This will result in rejection of the other body locations. Similarly, for sit to stand and stand to sit, the other body segments do not have major rotation along the gravity, and it results in the poor precision.

In Figure 10(c), for the arm lateral raise (left) activity, the left upper arm (#3) and left forearm (#4) have the same precision of 50%. During this activity, the forearm and upper arm have the same rotations along the gravity vector, and they are the only false positive for each other. For the other six body locations, there are no motions during this activity, and they are the false positives for others. The activity arm lateral raise (right) has the same explanation. For the pick-up phone activity, the precision for the forearm (#2 or #4) and upper arm (#1 or #3) are both 100%. It means this activity will determine the sensor location of the upper arm and forearm without any false positives.

As discussed in Section 4.3, our algorithm uses a cascade classifier to continuously reject the non-target locations until only one remains, and it will be determined as the target location. All the simple activities in this section can only construct one classifier and cannot offer the final classification results. To explore the ability of the cascade classifier, a varying number of sequences of simple activities are cascaded manually, and the average precisions are analyzed. Since all the simple activities offer 100% recall, any combination of them will also offer 100% recall.

Figure 11 shows the average precision for different numbers of combined simple activities. The X-axis is the number of activities combined and Y-axis is the average precision. Different lines represent different body locations. From the figure, we can see that the precision changes for different locations are very consistent. Also, a combination of 5 activities will achieve about 50% precision for all locations, which means there is only one other location confused with the target location. A very sharp slope is observed from #10 to #11. This means that when the activities increase from 10 to 11,

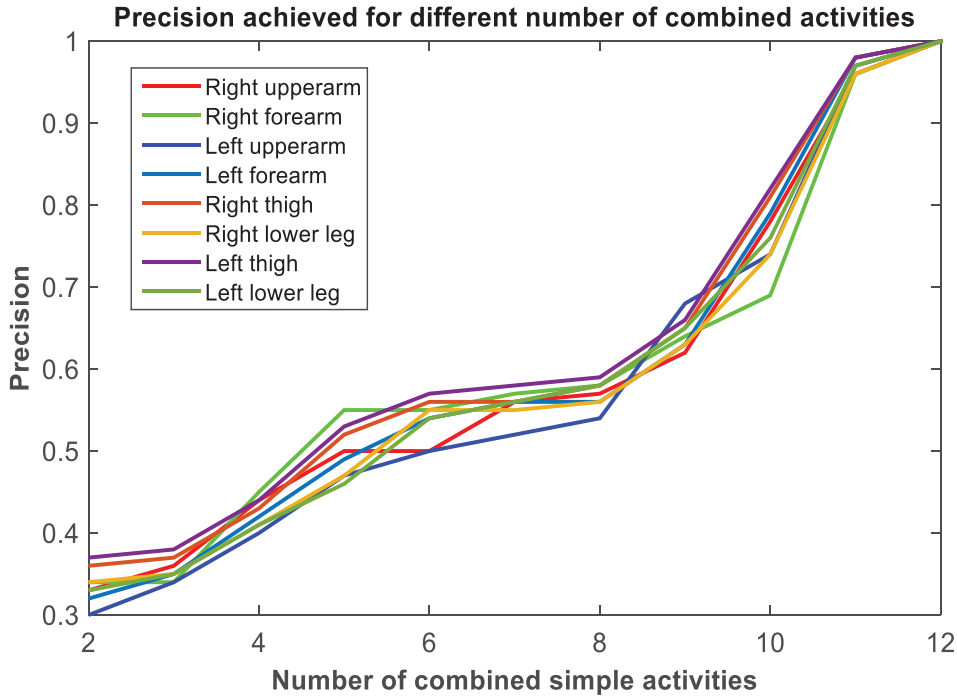

Fig. 11. Average precision for difference number of combined simple activities.

there is very large increase in precision. Also, if all 12 simple activities are combined, all locations will get 100% precision, which proves our algorithm will work well as long as enough good movements are provided.

6.2. Results of Complicated Daily Motion Tasks

To further evaluate how our algorithm performs for complicated motion tasks, each subject is asked to perform two complicated tasks (cooking and playing basketball) for 3min for each. They repeat each task 3 times. For this experiment, recall is used to evaluate the performance of our algorithm. The reason why we look at recall is that for a final decision, either it is correct or not correct. For correct instances, we consider them as true positives and for incorrect instances, we consider them as false negatives. Therefore, the recall measures the ratio between the correct recognized instances and all the instances belong to this location.

Figure 12 shows the recall of different body locations for cooking and playing basketball averaged on five subjects. From the figure, we can see that our algorithm achieves good performance for both cooking and playing basketball. The average recall for both tasks is 82.56%. For the cooking task, the recalls for the upper body locations (right upper arm, right forearm, left upper arm, and left forearm) are slightly bigger than the recalls for the lower body locations (right thigh, right lower leg, left thigh, and left lower leg). This is because, for the cooking task, there is more motion involvement from arms than from thighs and lower legs, and this will result in better discriminative performance. For playing basketball, our algorithm offers similar performance for both upper body locations and lower body locations, since they are both involved in a large amount of motion. Also, it is observed from the figure that our algorithm performs better for cooking than for playing basketball. The main reason is that the

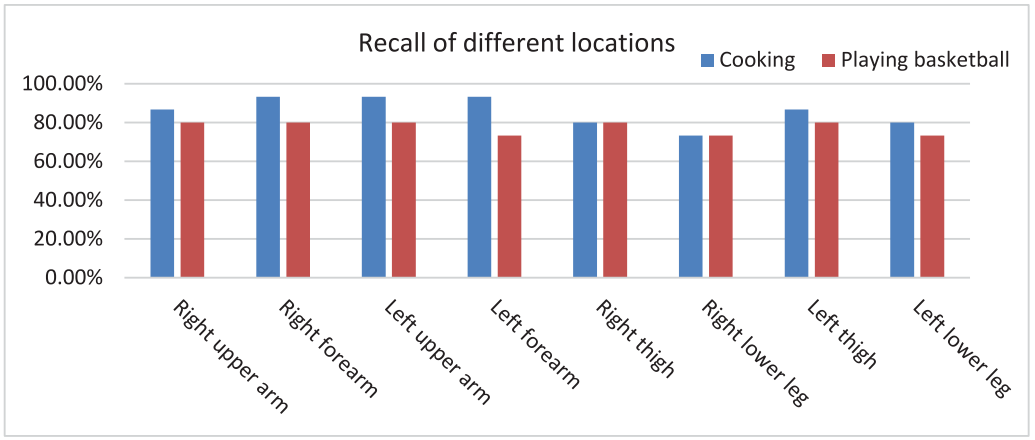

Fig. 12. Recall of different locations for cooking and playing basketball.

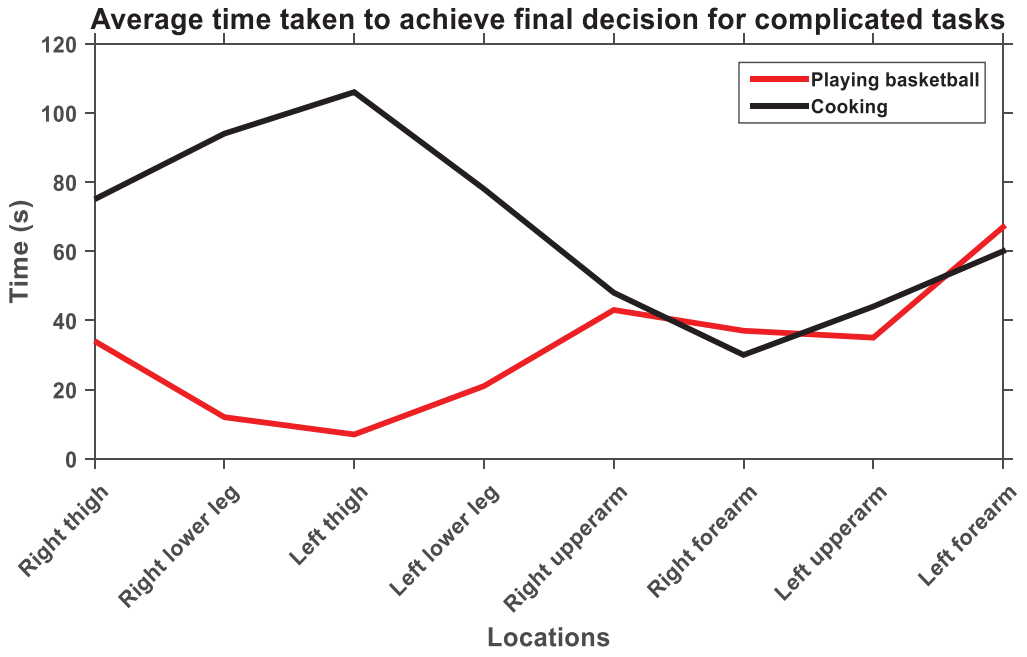

Fig. 13. Average total time taken for cascade classifier to achieve final decision.

motion involved in basketball is much faster than it is in cooking. Our weighting method partly addresses this issue, but if there are too many bad samples, it will lead to slightly poorer performance.

Besides the final recall, we also look at how long it takes for the cascade classifier to achieve the final decision. Figure 13 shows the time taken for the cascade classifier to achieve the final decision for different locations of these two different complicated motion tasks. The time is averaged from five subjects. From the figure, we can see that for playing basketball, the classifier achieves a final decision for all locations in less than 60s. This is because playing basketball consists of a lot of motions that can be used

to reject non-target locations at this decision node. Although the performance is worse than cooking, as discussed in Figure 12, the time taken to achieve the final decision is less than that of cooking. For cooking, it takes longer for the four segments of the lower body to achieve a final decision. The reason is that, for the cooking experiments, there is much more upper body motion than lower body motion, which makes it easier and faster to determine the sensor locations on the upper body.

7. CONCLUSIONS AND DISCUSSION

In this article, we proposed an effortless vision-assisted localization and calibration technique for wearable inertial sensors. A cascade decision-tree-based classifier determines the on-body sensor locations based on the least-squares errors obtained by solving the Wahba's problem between accelerometer and Kinect skeleton segment vectors. Our proposed weighting adjusting scheme and the vision occlusion consideration ensure that our approach operates robustly. We evaluate our approach with two experiments: simple daily activities and complicated motion tasks. Our approach achieves 100% recall for simple actions and 82.56% recall for complicated motion tasks.

Wearable devices are important parts of IoT devices, and the on-body placement of wearable devices is important to develop robust algorithms for different applications. The calibration of the placement of wearables is important and should be accomplished seamlessly. Fortunately, IoT settings provide opportunities to fuse information from environmental and wearable sensors. These opportunistic fusion techniques enable the calibration of one modality using other modalities. In our article, camera and vision-based IoT devices help calibrate the placement of wearables. However, the same idea can be explored between other modalities of devices. How diverse sensors can benefit from similar techniques would be an interesting topic to investigate.

In our article, we assume the wearable data and vision device data can be extracted for the same person and processed in the cloud. However, in reality, if multiple individuals are appearing in front of the camera, then challenges associated with determining who should be assigned to which wearable set must be addressed. Although this is out of scope of our article, a possible solution for the extension of Wahba's is to include a larger number of sensors and body segments from multiple users.

REFERENCES

G. Bahle, P. Lukowicz, K. Kunze, and K. Kise. 2013. I see you: How to improve wearable activity recognition by leveraging information from environmental cameras. In *Proceedings of the 2013 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*. IEEE, 409–412.

R. Bartlett. 2007. *Introduction to Sports Biomechanics: Analysing Human Movement Patterns*. Routledge.

T. R. Bennett, C. Savaglio, D. Lu, H. Massey, X. Wang, J. Wu, and R. Jafari. 2014. Motionsynthesis toolset (most): A toolset for human motion data synthesis and validation. In *Proceedings of the 4th ACM MobiHoc Workshop on Pervasive Wireless Healthcare.* ACM, 25–30.

D. J. Berndt and J. Clifford. 1994. Using dynamic time warping to find patterns in time series. In *Proceedings of the KDD Workshop*. Seattle, WA. 10, 359–370.

Y. J. Chang, S. F. Chen, and J. D. Huang. 2011. *A Kinect-based system for physical rehabilitation: A pilot study for young adults with motor disabilities. Res. Dev. Disabil.* 32, 6, 2566–2570.

Y. L. Chen. 2001. Application of tilt sensors in human-computer mouse interface for people with disabilities. *IEEE Trans. Neur. Syst. Rehabil. Eng.* 9, 3 (2001), 289–294.

J. L. Crassidis, K. L. Lai, and R. R. Harman. 2005. Real-time attitude-independent three-axis magnetometer calibration. *J. Guid. Control Dynam.* 28, 1 (2005), 115–120.

H. Ghasemzadeh and R. Jafari. 2011. Physical movement monitoring using body sensor networks: A phonological approach to construct spatial decision trees. *IEEE Trans. Industr. Inform.* 7, 1 (2011), 66–77.

P. D. Groves. 2013. *Principles of GNSS, Inertial, And Multisensor Integrated Navigation Systems.* Artech House.

T. Helten, M. Muller, H. P. Seidel, and C. Theobalt. 2013. Real-time body tracking with one depth camera and inertial sensors. In *Proceedings of the 2013 IEEE International Conference on Computer Vision (ICCV).* IEEE, 1105–1112.

K. Hung, Y. Zhang, and B. Tai. 2004. Wearable medical devices for tele-home healthcare. In *Proceedings of the 26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (IEMBS'04).* IEEE, 5384–5387.

E. Jovanov, A. Milenkovic, C. Otto, and P. C. De Groen. 2005. A wireless body area network of intelligent motion sensors for computer assisted physical rehabilitation. *J. NeuroEng. Rehabil.* 2, 1 (2005), 6.

J. Kjeldskov and C. Graham. 2003. A review of mobile HCI research methods. In *Human-computer Interaction with Mobile Devices and Services*. Springer, 317–335.

K. Kunze and P. Lukowicz. 2007. Using acceleration signatures from everyday activities for on-body device location. In *Proceedings of the 2007 11th IEEE International Symposium on Wearable Computers.* IEEE, 115–116.

K. Kunze, P. Lukowicz, H. Junker, and G. Tröster. 2005. Where am i: Recognizing on-body positions of wearable sensors. In *Location-and Context-awareness*. Springer, 264–275.

B. Lange, C. Y. Chang, E. Suma, B. Newman, A. S. Rizzo, and M. Bolas. 2011. Development and evaluation of low cost game-based balance rehabilitation tool using the Microsoft Kinect sensor. In *Proceedings of the 2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC).* IEEE, 1831–1834.

E. A. Lee, J. Rabaey, B. Hartmann, J. Kubiatowicz, K. Pister, A. Sangiovanni-Vincentelli, S. A. Seshia, J. Wawrzynek, D. Wessel, and T. S. Rosing. 2014. The swarm at the edge of the cloud. *IEEE Des. Test* 31, 3 (2014), 8–20.

G. M. Lerner. 1978. Three-axis attitude determination. *Spacecr. Attitude Determ. Control* 73 (1978), 420–428.

R. Lienhart, L. Liang, and A. Kuranov. 2003. A detector tree of boosted classifiers for real-time object detection and tracking. In *Proceedings of the 2003 International Conference on Multimedia and Expo (ICME'03).* IEEE, II–277.

S. Madgwick. 2010. An efficient orientation filter for inertial and inertial/magnetic sensor arrays. Report x-io and University of Bristol (UK).

D. Mizell. 2003. Using gravity to estimate accelerometer orientation. In *Null.* IEEE, 252.

B. Najafi, K. Aminian, A. Paraschiv-Ionescu, F. Loew, C. J. Bula, and P. Robert. 2003. Ambulatory system for human motion analysis using a kinematic sensor: monitoring of daily physical activity in the elderly. *IEEE Trans. Biomed. Eng.* 50, 6 (2003), 711–723.

I. Oikonomidis, N. Kyriazis, and A. A. Argyros. 2011. Efficient model-based 3d tracking of hand articulations using kinect. In *BMVC*, 1, 2, 3.

I. Oikonomidis, N. Kyriazis, and A. A. Argyros. 2012. Tracking the articulated motion of two strongly interacting hands. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* IEEE, 1862–1869.

G. Pons-Moll, A. Baak, T. Helten, M. Muller, H. P. Seidel, and B. Rosenhahn. 2010. Multisensor-fusion for 3d full-body human motion capture. In *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* IEEE, 663–670.

D. M. W. Powers. 2011. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. *Journal of Machine Learning Technologies* 2, 1 (2011), 37–63.

D. Roggen, K. Förster, A. Calatroni, T. Holleczek, Y. Fang, G. Tröster, P. Lukowicz, G. Pirkl, D. Bannach, and K. Kunze. 2009. Opportunity: Towards opportunistic activity and context recognition systems. In *Proceedings of the 2009 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks and Workshops (WoWMoM'09)*. IEEE, 1–6.

M. E. Sargin, Y. Yemez, and E. Erzin. 2007. Audiovisual synchronization and fusion using canonical correlation analysis. *IEEE Trans. Multimed.* 9, 7 (2007), 1396–1403.

P. G. Savage. 1998. Strapdown inertial navigation integration algorithm design part 1: Attitude algorithms. *J. Guid. Control Dynam.* 21, 1 (1998), 19–28.

A. Vahdatpour, N. Amini, and M. Sarrafzadeh. 2011. On-body device localization for health and medical monitoring applications. In *Proceedings of the 2011 IEEE International Conference on Pervasive Computing and Communications (PerCom).* IEEE, 37–44.

P. Viola and M. J. Jones. 2004. Robust real-time face detection. *Int. J. Comput. Vis.* 57, 2 (2004), 137–154.

G. Wahba. 1965. A least squares estimate of satellite attitude. *SIAM Rev.* 7, 3 (1965), 409–409.

D. Weenk, B. J. F. Van Beijnum, C. T. Baten, H. J. Hermens, and P. H. Veltink. 2013. Automatic identification of inertial sensor placement on human body segments during walking. *J. Neuroeng. Rehabil.* 10, 1 (2013), 31.

J. Wu, Z. Wang, S. Raghuraman, B. Prabhakaran, and R. Jafari. 2014. Demonstration abstract: Upper body motion capture system using inertial sensors. In *Proceedings of the 13th International Symposium on Information Processing in Sensor Networks (IPSN-14).* IEEE, 351–352.

Received December 2015; revised July 2016; accepted November 2016