Zero-Effort Camera-Assisted Calibration Techniques for Wearable Motion Sensors

Jian Wu and Roozbeh Jafari
The University of Texas at Dallas, Richardson, Texas, 75080
{jian.wu, rjafari}@utdallas.edu

ABSTRACT
Activity recognition using wearable motion sensors plays an important role in pervasive wellness and healthcare monitoring applications. The activity recognition algorithms are often designed to work with a *known* orientation of sensors on the body. In the case of accidental displacement of the motion sensors, it is important to identify the new sensor location and orientation. This step, often called *calibration* or *recalibration*, requires extra effort from the user to either perform a set of known movements, or enter information about the placement of the sensors manually. In this paper, we propose a camera-assisted calibration approach that does not require any extra effort from the user. The calibration is done seamlessly when the user appears in front of the camera (in our case, a Kinect camera) and performs an arbitrary activity of choice (*e.g.*, walking in front of the camera). We provide experimental results supporting the effectiveness of our approach.

Keywords
Inertial wearable sensors, Kinect cameras, Zero-effort calibration, Activity recognition

Categories and Subject Descriptors
H.4.0. Information System Application: General

1. INTRODUCTION
The recognition of activities of daily living (ADLs) has attracted a lot of attention in recent years due to its importance for healthcare and wellness monitoring applications [1, 2]. Wearable sensors capable of monitoring activities of daily living are gaining popularity due to their minimal cost, ubiquitous nature and ability to provide sensing functionality at any time and place. Activity recognition algorithms typically assume the configuration (*e.g.*, sensor location and orientation) of the sensors is known and does not change throughout the deployment. However, accidental displacement of the sensors may occur due to the user's movements. Moreover, there is no guarantee that the user will place the sensors at the expected orientation and location as required by the activity recognition algorithms. These factors will affect movement detection accuracy. Therefore techniques to track accidental misplacement and movements of the sensors are required.

To address issues associated with misplacement of the sensor, three approaches have been proposed in the prior studies. The first approach focuses on selecting features that are not affected by the sensor displacement [3]. The main problem with this approach is that the pre-determined feature space suitable for this approach typically limits the functionality, accuracy and the performance of signal processing algorithms. The second approach attempts to provide statistical techniques to adjust the feature space adaptively in order to improve the accuracy of the recognition algorithm in the presence of accidental misplacement [4]. The disadvantage of this approach is somewhat limited performance if a major displacement or orientation change occurs. The third approach is based on determining the exact displacement and orientation change that occurred and utilizing rotation and translation techniques to convert the raw sensor readings into the desired space and orientation [5]. With this approach, the raw sensor readings are transformed to appear as if the sensors were not misplaced, allowing the original activity recognition algorithms to continue to operate efficiently. The approach proposed in this paper belongs to the third category.

In this paper, we propose a novel approach to detect the orientation of the sensors leveraging the capability of a 3-D Kinect camera, when available, and perform the calibration step seamlessly without requiring extra effort from the user. We illustrate the capability of our proposed approach using an example: John wears his music player on his arm. The music player has a motion sensor. The sensor is intended to monitor his physical activity when he goes to the gym. When he enters the gym, a 3-D Kinect camera placed at the door captures his skeletal information during walking, and if his consent is provided, this is shared via the cloud. Our proposed technique uses the Kinect stream along with the readings acquired from his sensor to identify the orientation of the sensor on his arm. The core technical contribution of this work includes: 1) proposing a two-step search algorithm that calculates the sensor orientation with respect to (w.r.t.) the human body frame based on rotation distance. The human body frame and rotation distance are defined in Section 3.3 and 4.2 respectively; 2) the calibration method does not require extra effort from the user; 3) In our algorithm, the orientation of the inertial sensor w.r.t. the Kinect frame is calculated. This is very important for some applications in which the frames of the inertial sensor and Kinect cameras need to be aligned and used together.

In the remainder of the paper, we first provide an overview of the prior art in Section 2. We provide preliminary information on wearable motion sensors and the Kinect along with the problem formulation in Section 3. In Section 4, we introduce a two-step zero-effort search algorithm to determine the sensor orientation w.r.t. the human body frame. In Section 5, we provide our experimental studies validating the proposed technique, followed by the conclusion.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
Wireless Health '14, Oct 29-31, 2014, Bethesda, MD, USA.
Copyright 2014 ACM 978-1-4503-3160-9 ... $15

2. BACKGROUND AND RELATED WORK

It is known that sensor displacement affects the accuracy of activity recognition algorithms. The impact of the sensor translation and rotation on a sample activity recognition algorithm called dynamic time warping is discussed in [6]. In [7], the authors explore how the rotation and translation displacement affect the results of recognition algorithms and provide recommendations about how to deal with sensor displacement.

Several approaches have been proposed to address the issue associated with sensor displacement and their impact on recognition algorithms. A solution to find the displacement invariant features has been proposed. In [3], device orientation independent features are used for step detection and direction estimation for the applications of dead reckoning. Another approach to study the statistical distribution of the features, and adjust the features adaptively has been suggested. The possibility of system self-calibration through the adjustment of the classifier decision boundaries is proposed in [8]. Similarly in [4], the authors propose a method to compensate for the data distribution shift caused by sensor displacements using an expectation-maximization algorithm and covariance shift analysis. These approaches adjust the feature space in case of a minor displacement. However, they cannot calibrate the presence of substantial displacements. If a major displacement occurs, the recognition algorithm will function with poor accuracy.

Other researchers focus on identifying the exact orientation and translation change during the movement by asking the user to perform certain movements. For instance, some have analyzed the acceleration data during walking to determine the sensor placement [9] and orientation [10]. In [5], the authors defined a global frame by asking the user to perform symmetric forward/backward movements (*e.g.,* walking). The accelerometer signals are projected to this global frame such that the sensor orientation is calibrated to this global frame. The disadvantage of this approach is that the user would be required to perform certain specific movements and if the global frame is altered, the algorithm will likely fail.

3. PRELIMINARIES

(a). Kinect sensor. (b). Inertial sensor.

Figure 1. Kinect sensor and motion sensor.

3.1 Kinect and Inertial Sensor

Kinect, as shown in Figure 1.a, is a low-cost RGB-Depth sensor introduced by Microsoft. Two libraries (Kinect SDK and Open NI/NITE) are available that provide access to the depth map or the skeleton tracking information. In this paper, we use Kinect SDK to get the absolute position data of each joint of the human body. Two joints will be used as the end points to construct a segment vector and the rotation of this vector represents the rotation of this body segment in the Kinect frame. This information is used in our approach. Figure 1.b shows the 9-axis motion sensor with dimensions of 1"×1.5" that measures 3-axis acceleration, 3-axis angular velocity and 3-axis magnetic field. The data can be streamed to a PC/tablet for real time signal processing, storage or can be logged onto a MicroSD card.

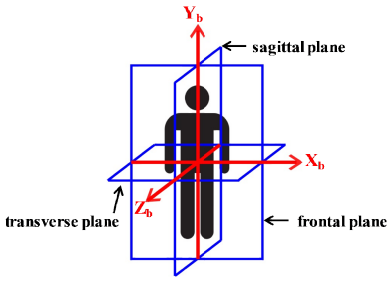

Figure 2. Human body frame (back view).

3.2 Frame Definitions

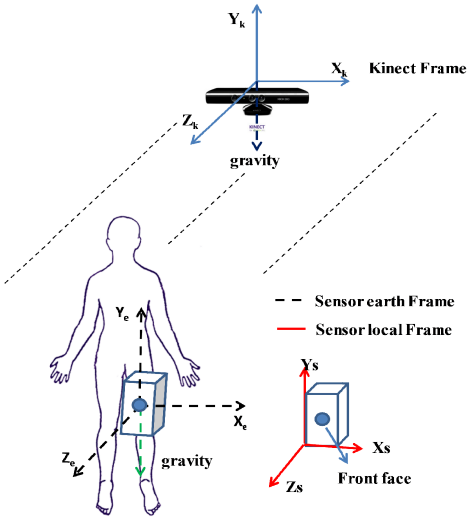

Figure 3. Frame definitions.

There are four coordinate frames in our paper: the human body frame, the sensor local frame, the sensor earth frame and the Kinect frame. These frames are defined as:

1). *Human body frame*: the human body frame is defined in Figure 2. It is the back view of a human body. The axes are represented by X_b, Y_b and Z_b.

2). *Sensor local frame*: The sensor local frame is shown in Figure 3. The face with a circle is the front face of the sensor. The Z_s always points out of the front face and X_s and Y_s are parallel to two sensor edges as shown in the figure.

3). *Sensor earth frame*: Sensor earth frame is defined as the blue dashed lines in Figure 3. The positive *Y*-axis Y_e is in the opposite direction of gravity. The positive *Z*-axis Z_e is parallel to the ground and is the projection of the sensor local frame *Z-axis* onto the transverse plane. The *X*-axis X_e is uniquely defined by Y_e and Z_e of a right handed orthogonal coordinate. Notice that the sensor may face in any direction, so the sensor earth frame is not unique and is determined by the direction of projection of the sensor local frame *Z-axis* onto the transverse plane.

4). *Kinect frame*: The Kinect frame coordinates are shown on the Kinect in Figure 3. The positive *Z*-axis Z_k points in the direction in which the Kinect is facing. The positive *Y*-axis Y_k is upward and the positive *X*-axis X_k points to the right of the Kinect (when

viewed from the front). The tilt of the Kinect is 0 degrees, so Y_k is in the opposite direction of gravity.

3.3 Problem Formulation

The problem of the sensor displacement on rigid human body segments is divided into three sub problems in this paper as shown in Figure 4. The cylinders represent the human body segments (*e.g.* leg or arm), which have the same reference frame as the human body frame. The first case is the sensor's rotational displacement around Y-axis of human body frame, which is denoted as *yaw rotation* in this paper. The second case is the rotation of the sensor along the Z-axis of the human body frame which is called *roll rotation*. The third case is the sensor's line displacement along the segment which is denoted as sensor translation. The literature has shown that the impact of case 3 is often negligible on signal processing algorithms [6]. In this paper, we only focus on the calibration of the sensor yaw rotation β (case 2) and roll rotation γ (case 1) w.r.t. the human body frame.

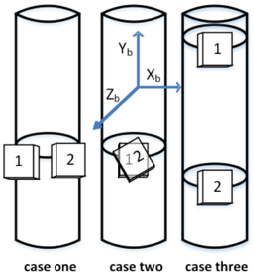

Figure 4. Three cases of sensor displacement.

To simplify the problem, we first assume the user faces Kinect camera such that the *Z*-axis of human body frame points in the same direction as the *Z*-axis of the Kinect frame. Since the human body frame is oriented the same as the Kinect frame in this scenario, the problem becomes determining the sensor yaw rotation -β and roll rotation, γ, w.r.t. the Kinect frame. Once we obtain the results, we relax the assumption so that the human can face anywhere. The yaw rotation α between the Kinect frame and human body frame, which is the rotation about *Y-axis*, can be obtained from Kinect API. Now we can calculate the yaw rotation of sensor frame w.r.t. the human body frame. Since the *Z-axis* of the Kinect frame and the human body frame are in the same plane (human transverse plane), the sensor roll rotation w.r.t. the human body frame is the same as the sensor roll rotation w.r.t. the Kinect frame.The *Z-axis* of the sensor earth frame is defined as the direction of the projection of the sensor front face in the transverse plane of the human body frame, the yaw rotation of sensor local frame w.r.t. the Kinect frame is the same as the yaw rotation of the sensor earth frame w.r.t. the Kinect frame, which is a rotation along Kinect *Y-axis*. The angles used in this paper are defined in Table 1.

Table 1. Angle definitions.

Angle symbol	Angle representation
β	Yaw rotation of the Kinect frame w.r.t. the sensor local frame
γ	roll rotation of the sensor local frame w.r.t. the Kinect frame
α	Yaw rotation of the Kinect frame w.r.t. the human body frame
φ	Yaw rotation of the sensor local frame w.r.t. the human body frame

4. METHODOLOGY

In this section, we first explain the orientation filter that estimates the orientation of sensor local frame w.r.t. the sensor earth frame. We also introduce the rotation distance between 3-D rotations. Next, the two-step search algorithm is explained. In the first step, a body segment rotation is measured in the sensor earth frame. Meanwhile, the same rotation is measured from Kinect skeleton data after rotating the Kinect frame into the sensor earth frame. By searching for the minimum rotation distance between the rotation measured from inertial sensor and the Kinect, the yaw rotation β between the sensor earth frame and the Kinect frame is determined. The same approach is applied to the second step search to determine the sensor roll rotation w.r.t. the Kinect frame γ. The yaw rotation between the Kinect frame and the human body frame, α, can be obtained from the Kinect API. As we have the yaw rotation between the sensor local frame and the Kinect frame and the yaw rotation between the Kinect frame and the human body frame, the yaw rotation of the sensor local frame w.r.t. the human body frame φ is achieved. Since the *Z-axis* of the Kinect frame and human body frame are in the parallel planes (parallel to human transverse plane), the sensor roll rotation w.r.t. the human body frame is the same as the sensor roll rotation w.r.t. the Kinect frame. Thus, the sensor yaw rotation and roll rotation w.r.t. the human body frames are both calibrated.

4.1 Orientation Estimation for Inertial Sensor

The inertial sensor measures 3-axis acceleration and 3-axis angular velocity in its local frame. An orientation filter [11] is used to estimate the orientation of the sensor local frame w.r.t. the sensor earth frame, which is denoted as q_S^E. q is a quaternion representation of the orientation. S represents sensor local frame and E represents the sensor earth frame. A quaternion q is a four-dimensional complex number ($[q_w\ q_x\ q_y\ q_z]$) that can be used to represent the orientation of a coordinate frame in 3-D space. The relationship between quaternion and a rotation is explained in Section 4.3.

4.2 Rotation Distance

Before introducing our two-step search algorithm, we briefly discuss rotation distance metrics which are important measurements throughout our algorithm deployment. 3D rotations are widely used in numerous applications such as computer vision, computer graphics and robotics. The evaluation of the distance between two 3D rotations is an important task. There are several rotation distance metrics in the literature based on Euler angles, unit quaternion and rotation matrices [12], which are commonly used to represent 3D rotations. The 3D rotation matrix is a 3×3 orthogonal matrix that is used to perform a rotation in Euclidean space. The matrices form a group called the Special Orthogonal group, SO(3), in which the operations of multiplication and inversion are continuous functions of the matrix entries. In this paper, we use a rotation metric introduced in [13] to derive our algorithm. The metric is denoted as

$$\mathrm{d}(\mathrm{R}_1, \mathrm{R}_2) = \mathrm{tr}(\mathrm{I} - \mathrm{R}_1\mathrm{R}_2^{\mathrm{T}}) \quad (1)$$

where R_1 and R_2 are two rotation matrices, I is an identity matrix and tr represents the trace of a matrix.

4.3 Two-step Search Algorithm

4.3.1 First step yaw rotation search

The first step search finds the *yaw rotation* of the sensor local frame w.r.t. the Kinect frame, which is the same as the yaw

rotation of the sensor earth frame w.r.t. the Kinect frame. As shown in Figure 5, X_e, Y_e and Z_e are three axes of the sensor earth frame and X_k, Y_k and Z_k are three axes of the Kinect frame. Y and Y_k are parallel to each other and point to the opposite of the gravity vector. The yaw rotation β is the rotation along Y axis from Kinect frame to sensor earth frame.

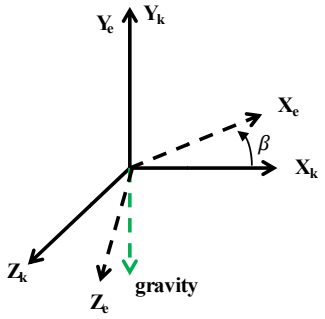

Figure 5. Yaw rotation formulation.

Consider the rotation of the body segment AB in Figure 6, this rotation can be measured from the inertial sensor in the sensor earth frame or from the Kinect skeleton joint position information in the Kinect frame. If the two frames are rotated into the same reference frame, the measured rotations from these two systems should be the same. In this paper, we rotate the Kinect frame to the sensor earth frame.

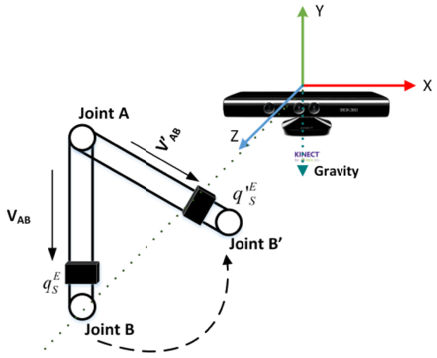

Figure 6. Body segment rotation.

First, we calculate the rotation of body segment AB in the sensor earth frame. As discussed in Section 4.1, the orientation filter can measure the sensor orientation w.r.t. the sensor earth frame, and q_S^E and q'^E_S represent the sensor orientation w.r.t. the sensor earth frame at two different states of the body segment movement. The 'state' represents the state of the body segment at a certain time. For example, for a sit-to-stand movement, one state would be the orientation of body segment (*e.g.* thigh) in sitting posture and the other state would be the orientation of the body segment in standing posture. The segment rotation in the sensor earth frame is calculated by:

$$q_E = (q_S^E)^{-1} \otimes q'^E_S \tag{2}$$

q_E is the segment rotation in sensor earth frame and $(q_S^E)^{-1}$ is the inverse of q_S^E. The inverse of a quaternion, q^{-1} , represents the inverse rotation of q. $\otimes$ represents quaternion multiplication.

The rotation of segment AB is then constructed from Kinect position data. The position data for joint A and joint B can be obtained from the Kinect API for every frame, denoted as P_A and P_B respectively. A 5 span moving average filter is applied to the position data to smooth the noise. V_{AB} and V'_{AB} are the body segment vectors at states AB and AB'. They are defined as:

$$V_{AB} = P_B - P_A \tag{3}$$

$$V'_{AB} = P'_B - P_A \tag{4}$$

If these two vectors are rotated to the sensor earth frame from the Kinect frame, they can be used to construct the segment rotation in sensor earth frame. The rotation from Kinect frame to sensor earth frame is represented by a rotation axis [0, 1, 0] (*Y-axis*) and a rotation angle β. The rotation axis is the axis about which the rotation happens. The representation of the rotation by rotation axis vector **r** and rotation angle θ can be translated to a quaternion representation q by:

$$\left.\begin{aligned} q_x &= r_x * \sin(\theta/2) \\ q_y &= r_y * \sin(\theta/2) \\ q_z &= r_z * \sin(\theta/2) \\ q_w &= \cos(\theta/2) \end{aligned}\right\} \tag{5}$$

qx, qy, qz and qw are the four components of the quaternion vector q and rx, ry and rz are the three components of the unit vector $\boldsymbol{r}$. The Kinect frame w.r.t. the sensor world frame $q_K^E(\beta)$ can then be calculated by (5). The 3-D vectors V_{AB} and V'_{AB} are rotated into the sensor earth frame as $V_{AB}^E(\beta)$ and $V_{AB}'^E(\beta)$ by:

$$[\,0 \quad V_{AB}^E(\beta)] = (q_K^E(\beta))^{-1} \otimes [0 \quad V_{AB}] \otimes q_K^E(\beta) \tag{6}$$

$$[\,0 \quad V_{AB}'^E(\beta)] = \left(q_K^E(\beta)\right)^{-1} \otimes [0 \quad V'_{AB}] \otimes q_K^E(\beta) \tag{7}$$

Zero in (6) and (7) is value of qw of a quaternion. The rotation from $V_{AB}^E(\beta)$ to $V_{AB}'^E(\beta)$ can be represented by a rotation axis **r** and the rotation angle θ. The axis **r** and rotation angle θ can be achieved by:

$$V_{AB}^E(\beta) \cdot V_{AB}'^E(\beta) = \left\|V_{AB}^E(\beta)\right\| \left\|V_{AB}'^E(\beta)\right\| \cos\theta \tag{8}$$

$$V_{AB}^E(\beta) \times V_{AB}'^E(\beta) = \left\|V_{AB}^E(\beta)\right\| \left\|V_{AB}'^E(\beta)\right\| \sin\theta\,\mathbf{r} \tag{9}$$

r and θ are also functions of β, after normalizing **r**, the segment rotation in the sensor earth frame constructed from two segment vectors $q'_E(\beta)$ is calculated by (5).

Now that q_E and $q'_E(\beta)$ represent the same rotation in the sensor earth frame, and thus the rotation distance between them should be 0. In reality, the Kinect skeleton vectors do not perform exactly the same as the inertial sensors. The inertial sensor has the rotation along the segment itself (*e.g.* the twist of the arm), but the Kinect will be unable to capture this degree of rotation. As a result, the rotation distance between q_E and $q'_E(\beta)$ is not exactly 0. If we search for β from 1 degree to 360 degrees, the targeted β will give the minimum rotation distance between q_E and $q'_E(\beta)$. In order to calculate the rotation distance, we first convert q_E and $q'_E(\beta)$ to rotation matrices R_E and $R'_E(\beta)$. The rotation distance $d(R_E, R'_E(\beta))$ can be calculated from (1).

Our search algorithm relies on the rotation between two states; if there is no rotation or a very small rotation, the movement will not give a good estimation of the yaw rotation angle β. Before beginning the search, we need to choose two states between which sufficient rotation occurs. It is important that we do not use the rotation between two states measured from inertial sensor to

check the rotation since the inertial sensor can have the rotation along the segment itself, which is different from the measurement of the Kinect system and cannot be used for our algorithm. In this paper, the rotation quality is validated from the segment rotation constructed from Kinect skeleton data. Since the gravity vector remains the same during one rotation of a body segment, the rotation between the body segment vector and the gravity vector q_{BS}^{G} is calculated for each frame during the rotation. By comparing the rotation of a body segment w.r.t. the gravity vector at two states, we can get the rotation distance of this body segment between the two states. For two different states during one body segment rotation, state 1 and state 2, we have q_{BS1}^{G} and q_{BS2}^{G}. Once we convert them to rotation matrices R_{BS1}^{G} and R_{BS2}^{G}, the rotation distance between state 1 and state 2, μ, can be calculated as:

$$\mu = d\left(R_{BS1}^{G}, R_{BS2}^{G}\right) \quad (10)$$

A rotation distance below 0.1 is considered to be too small to be used in our algorithm. The choice of threshold value of μ is discussed in Section 5.3.1. Based on the above analysis, the first step yaw search algorithm is formulated and described in Algorithm 1.

Algorithm 1 First step yaw direction search

Calculate μ according to (10);
  if $\mu < 0.1$
    The movement is not qualified, choose another one;
  else
    Continue;
  end
  for β= 1:360
    Calculate $d(R_E, R'_E(\beta))$;
    $\hat{\beta} = argmin(d(R_E, R'_E(\beta)))$;
  end
 return $\hat{\beta}$.

The $\hat{\beta}$ is the estimated yaw rotation of the Kinect frame w.r.t. the sensor local frame, so the yaw rotation of the sensor local frame w.r.t. the Kinect frame is -$\hat{\beta}$.

4.3.2 Second step roll rotation search

From the previous section, the estimated sensor yaw rotation -$\hat{\beta}$ w.r.t. the Kinect frame is obtained. In this section, we explain the second step search for the sensor roll rotation w.r.t. the sensor earth frame, which is the same as the sensor roll rotation w.r.t. the Kinect frame.

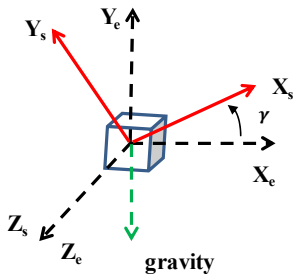

Figure 7. Roll rotation formulation.

Figure 7 shows the sensor roll rotation about the *Z-axis* of the sensor earth frame. *Xs*, *Ys* and *Zs* are the axes for the sensor local frame and X_e, Y_e and Z_e are axes for the sensor earth frame. Like the first step search, we define two states to calculate the same rotation measured by the Kinect skeleton and the inertial sensor in the sensor earth frame. One state is an ideal state in which the body segment vector is the same as the gravity vector and the inertial sensor only has a rotation of γ about the *Z-axis* of the sensor earth frame. The other state is an arbitrary state with the segment vector V_{AB}. The rotation between these two states is the rotation from the gravity vector to the body segment vector AB.

First, we construct the body segment rotation w.r.t. the gravity vector from the Kinect skeleton in the sensor earth frame. The yaw rotation $\hat{\beta}$ is known, and the vector V_{AB} can be rotated to the sensor earth frame by (6) as $V_{AB}^{E}(\hat{\beta})$. The gravity vector remains the same from the Kinect frame to the sensor earth frame and is represented by [0 -1 0]. Using (8), (9) and (5), the rotation between body segment vector and the gravity vector is calculated and represented by $q'_E(\hat{\beta})$.

Next, the rotation between two states from the inertial sensor is analyzed. The arbitrary state sensor orientation w.r.t. the earth frame is the output of the orientation filter q_S^E. At the ideal state, the sensor orientation w.r.t. the sensor earth frame is composed of the rotation axis *Z-axis* ([0 0 1]) and rotation angle γ. Then from (5), the sensor orientation w.r.t. the sensor earth frame at the ideal state is calculated as $q_{iS}^{E}(\gamma)$. The body segment rotation in sensor earth frame is defined as:

$$q_E(\gamma) = (q_{iS}^{E}(\gamma))^{-1} \otimes q_S^E \quad (11)$$

Covert $q'_E(\hat{\beta})$ and $q_E(\gamma)$ to rotation matrices $R'_E(\hat{\beta})$ and $R(\gamma)$. By searching for the minimum rotation distance $\Omega(\gamma)$ between $R'_E(\hat{\beta})$ and $R(\gamma)$, the γ is determined.

$$\Omega(\gamma) = \mathrm{d}(R'_E(\hat{\beta}), \mathrm{R}(\gamma)) \quad (12)$$

The second step of the search algorithm is described in Algorithm 2.

Algorithm 2 Second step roll rotation search

for γ = 1:360
    Calculate $\Omega(\gamma)$ according to (12);
    $\hat{\gamma} = argmin(\Omega(\gamma))$;
end
return $\hat{\gamma}$.

4.3.3 Sensor orientation with respect to the human body frame

From the two step search, we get the sensor roll and yaw rotation w.r.t. the Kinect frame. If the Kinect frame is the same as the human body frame (i.e., the subject is facing the Kinect), the roll and yaw are the same as w.r.t. the human body frame. However, if the user is not facing the Kinect such that the *Z-axis* of human body frame is not parallel to the *Z-axis* of the Kinect frame, there will be a yaw rotation between the Kinect frame and the human body frame. Notice that this will only affect the yaw rotation. Fortunately, we can get the yaw rotation between these two frames from Kinect API. The rotation of the hip center along Kinect *Y-axis* represents this yaw rotation. Let this yaw rotation be α, the sensor yaw rotation w.r.t. the human body frame is:

$$\varphi = -\hat{\beta} - \alpha \quad (13)$$

5. EXPERIMENTS AND RESULTS

5.1 Experimental Setup

In the experiments, the Kinect is placed on a flat table with a tilt angle 0 degrees. The subjects perform daily activities in front of

the Kinect. The subjects wear two sensors: one on the thigh and the other on the right upper arm. Four subjects were chosen for the experiments and 6 daily activities were performed to test our approach. The activities are listed in Table 2. These predefined activities are just some examples meant to exhibit generalizability of our method.

Table 2. Activities for the experiments.

No#	Activity	No#	Activity
1	Walking	4	Sit-to-Stand
2	Kneeling	5	Stand-to-Sit
3	Leg-Lifting	6	Arm Stretching

For each activity, the sensors are placed in four different yaw configurations approximately by the user. Figure 8 shows the four yaw configurations for thigh sensor: 0-degree in Figure 8.a, 90-degree in Figure 8.b, 180-degree in Figure 8.c and 270-degree in Figure 8.d. For each yaw configuration, the subjects were asked to place the sensors in two random roll rotations. All subjects repeated each activity 8 times.

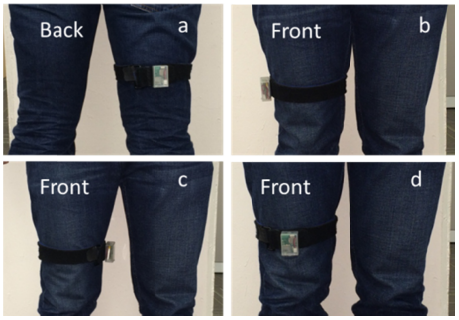

Figure 8. Four different yaw configurations.

Moreover, we compare the performance of our calibration method with a non-zero-effort method [5] using an activity recognition application.

5.2 Yaw Rotation Search

5.2.1 Choice of the threshold value of parameter μ

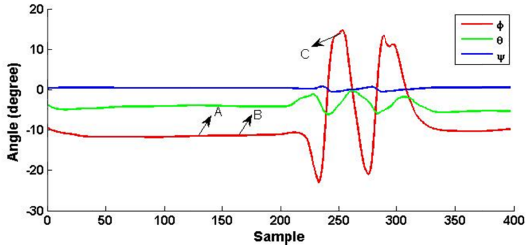

Figure 9. Euler angles between leg vector and gravity for walking.

As discussed in Section 4.3.2, the first step of the search algorithm is based on a body segment rotation. If there is no rotation or a very small rotation, the algorithm will not give an accurate yaw result. The parameter μ measures the rotation distance between two arbitrary states of the movement measured in the Kinect frame.

Figure 9 shows the Euler angles for a two-stride walking for one subject with a 0 degrees yaw configuration. The reason why we use Euler angles in this example is that they can give a better understanding on the amount of rotation during the movement. The Euler angles are calculated from the rotation of the body segment vector and the gravity vector in the Kinect frame from Kinect skeleton information. The *X-axis* in Figure 9 is the sample number for the sampling rate of 30 frames per second and the *Y-axis* is the value of the angular change. The red line, green line and the blue line represent the Euler angles along *X-axis*, *Y-axis* and *Z-axis*, respectively. For the first 200 samples, the subject was standing still and there is no angular change. After that, a two step-stride walking was performed and there is clear angular change along the *X-axis*. We select an arbitrary set of points called states. These points are marked with labels A, B and C in Figure 9. States A and B are chosen during the standing phase while there is no rotation. State C is chosen in the middle of the walking stride and shows a clear rotation from the previous two states (A and B). In practice, states can be selected arbitrarily. If no rotation is observed, and therefore the selected states would not be suitable, further states will be selected until an orientation change is observed.

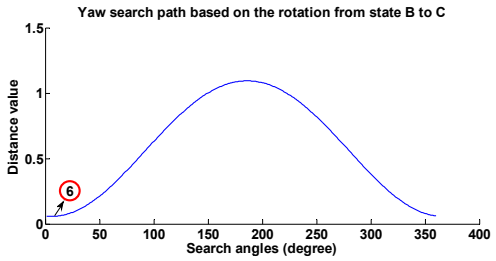

Figure 10. Path for first step search based on rotation from state B to state C.

Figure 10 shows the path for the first step search, which is based on the rotation from state B to state C. The X-axis represents the search angles, ranging from 1 degree to 360 degrees, and the Y-axis displays the rotation distance. The value '6', shown in the red circle in the figure, is the search angle that achieves the minimum rotation distance and thus is the result of our first step search algorithm. It is very close to 0 degrees, which is the approximate yaw configuration for this experiment. Also, the rotation distances between different search angles are distinguishable.

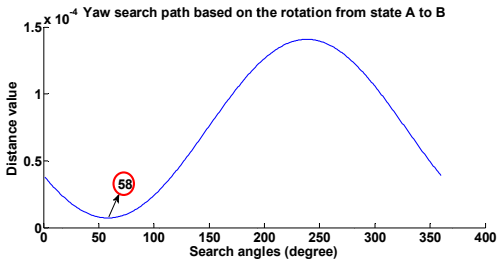

Figure 11. Path for first step search based on rotation from state A to state B.

Figure 11 shows the path for the first step search based on the rotation from state A to state B. From the figure, the search algorithm gives the result of 58 degrees which is incorrect for the 0-degree configuration. From the above analysis, we show that the search algorithm works correctly for the movement from state B to state C since there is a clear rotation while it does not work for the movement from state A to state B because of the small rotation. The rotation distance μ is 0.2022 between state B and

state C and is 1.4×10^{-4} between state A and state B for this walking case. To choose a suitable threshold for μ, we calculated μ for all the movements we used in the experiments and choose the minimum value of 0.1. This is a reliable measure since it works correctly for all our experiments. We will look into the optimal threshold for μ in the future.

5.2.2 *Results for yaw rotation search*

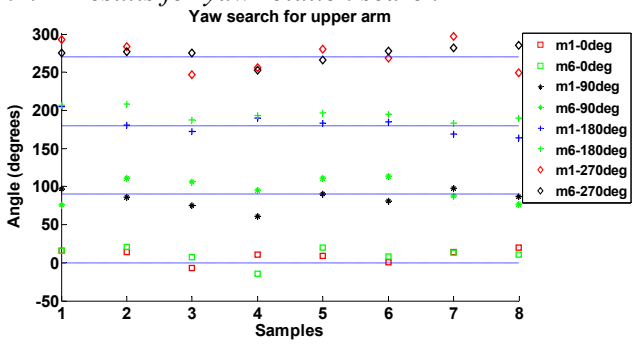

Figure 12. Search results for yaw rotation for upper arm.

To validate the yaw search technique, there should be a rotation that occurs during the movement. For the upper arm sensor, the segment has a qualified rotation only for arm stretching and walking, and thus only these two movements are chosen to calibrate the upper arm sensor. For the thigh sensor, all movements are used to calibrate the sensor except for the arm stretching. To test the yaw search technique, we manually place the sensor with the yaw rotation of 0 degrees, 90 degrees, 180 degrees and 270 degrees for all experimental configurations. Since these configurations are only approximately achieved by the user, they are not a gold standard. By checking the distribution of all the search results, we determine the consistence and the correctness of our algorithm. Figure 12 shows the search result distribution for the arm sensor. Since the arm only has sufficient rotation during movement 1 and movement 6, the results are reported only for these two movements. The X-axis represents the number of experiments and trials and the Y-axis represents the search results for yaw angles. The four dashed lines from the bottom to the top are the four expected yaw results $y = 0$, $y = 90$, $y = 180$ and $y = 270$, respectively. The label 'mx-ydeg' in the figure represents the results for movement x with a yaw configuration of y degrees. For example, 'm1-0deg' is the result for movement 1(walking) with the 0-degree yaw configuration. From the figure, we observe that for the two movements, all search results fall around the line which corresponds to their own yaw rotations. The distribution of the results proves our algorithm works well for the two arm sensor movements.

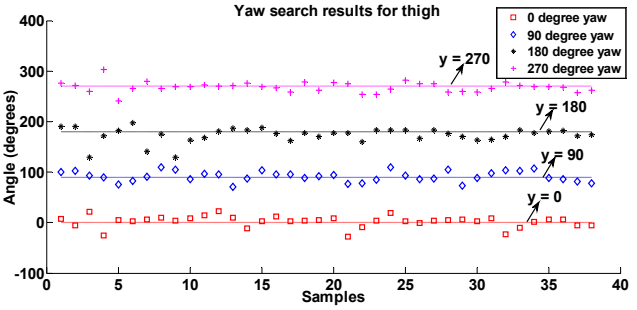

Figure 13. Search results for yaw rotation for thigh.

Figure 13 shows the distribution of all the search yaw results for the thigh sensor, for all 5 movements. The lines $y = 0$, $y = 90$, $y = 180$ and $y = 270$ are the four lines for the expected results for the configurations of 0 degrees, 90 degrees, 180 degrees and 270 degrees misplacement. The obtained results for the four configurations all fall near their corresponding lines. The results for the 180-degree configuration have larger deviations than the other three. The reason for this is that when we placed the sensor on the back of the subject; it is harder to find the 180 degree position than in the other three cases. The overall distribution in the figure shows good search results for the thigh sensor for all movements.

5.3 Second Step Roll Rotation Search

To validate the accuracy of our roll search algorithm, for each movement, we asked the user to stand still at the beginning of the movement and measured the roll rotation from the inertial sensor. The measured roll angle from inertial sensor serves as the ground truth of our algorithm because the orientation filter can achieve a very good accuracy [11] for the roll rotation calculation using the gravity as a reference.

Table 3. Roll search errors for different subjects for arm and thigh.

Subject #	Arm Accuracy (RMSE in degrees)	Thigh Accuracy (RMSE in degrees)
Subject 1	10.733	5.98
Subject 2	5.11	5.60
Subject 3	13.5	5.32
Subject 4	9.60	5.60
Total	10.73	5.59

Table 3 shows the root mean square errors between the inertial sensors reported roll results and the results obtained by our algorithm for different subjects. The thigh sensor has a 5.59 degree RMSE for all movements and for all the subjects while the arm sensor has a 10.73 degree RMSE. The reason for the difference is that the thigh has less moving freedom than the upper arm and the arm muscle movement will lead to more rotation along the segment than the thigh muscle which will affect the search algorithm. The accuracy of our calibration technique on thigh is very consistent (RMSEs between 5 and 6 degrees) for all subjects, which indicates that the thigh movement does not exhibit significant variations for different subjects. Conversely, the arm sensor accuracy varies from subject to subject because of the greater level of variations for the arm movements. The total RMSEs for both the arm and the thigh indicate that our algorithm achieves a good accuracy for the roll rotation.

5.4 Activity Recognition Performance

We also validate our approach leveraging an activity recognition application and compare it with the performance of a non-zero-effort approach [5]. Subjects perform the same set of activities with sensors with random orientations. A template matching algorithm based on dynamic time warping (DTW) is used to implement the recognition using 3-axis accelerations. The template is trained when the sensor is placed without any rotational displacement. During the testing phase, the accelerations are transformed to the original frame (prior to rotation displacement) and the transformed signals are compared

to the templates for classification. To determine the global frame as outlined in [5], the user is asked to perform 10 seconds of walking.

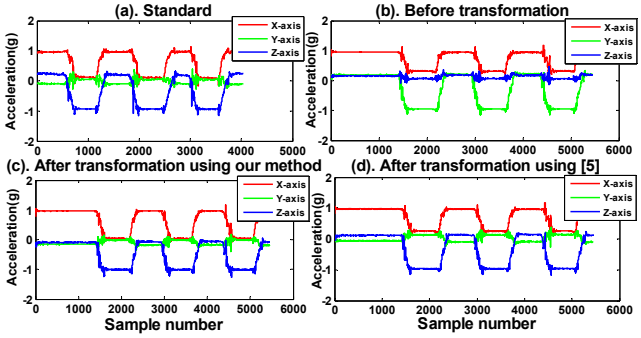

Figure 14. Calibration results for sit-to-stand and stand-to-sit patterns.

Figure 14 shows the calibration results for sit-to-stand and stand-to-sit patterns. Figure 14.a shows the standard pattern where no rotation displacement is present. Figure 14.b shows the signals before transformation while a rotation displacement is present and Figure 14.c and Figure 14.d show the calibrated patterns using our approach and using the approach in [5] respectively. It can be observed that both our method and the method in [5] have good calibration results. The calibrated signals from both approaches are similar to each other and are close to the standard signals.

Table 4. Activity recognition accuracy comparison

Method	Activity #					
	1	2	3	4	5	6
Our approach	93%	98%	92%	100%	93%	100%
Approach in [5]	92%	98%	90%	100%	92%	100%

Table 4 shows the performance of the activity recognition algorithm leveraging both calibration methods. We can see that both calibration methods achieve similarly high accuracy which illustrates the effectiveness of our zero-effort algorithm in conjunction with activity recognition. The approach in [5] performs slightly better than our approach likely due to the fact that the error from Kinect and skeleton construction likely impacts our calibration whereas vision-based sensors are not used in [5].

6. CONCLUSION

In this paper, we proposed a zero effort two-step search algorithm to calibrate orientation of wearable sensors by calculating the orientations of the sensors with respect to the human body frame based on rotation distance optimization. The experimental results from 4 subjects over 6 daily movements show that our algorithm achieves consistent and accurate results. We also evaluate the performance of our method for activity recognition and compare the results with a non-zero-effort approach and the results show our approach achieves similarly good performance.

7. ACKNOWLEDGMENT

This work was supported in part by the National Science Foundation, under grants CNS-1150079 and CNS-1012975, the National Institute of Health, under grant R15AG037971 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.

8. REFERENCES

[1] S. Mitchell, J. Collin, C. De Luca, A. Burrows, and L. Lipsitz, "Open-loop and closed-loop postural control mechanisms in parkinson's disease: increased mediolateral activity during quiet standing," *Neuroscience letters*, vol. 197, no. 2, pp. 133–136, 1995.

[2] J. Pansiot, D. Stoyanov, D. McIlwraith, B. P. Lo, and G.-Z. Yang, "Ambient and wearable sensor fusion for activity recognition in healthcare monitoring systems," in *4th international workshop on wearable and implantable body sensor networks (BSN 2007)*, pp. 208–212, Springer, 2007.

[3] U. Steinhoff and B. Schiele, "Dead reckoning from the pocket-an experimental study," in *Pervasive Computing and Communications (PerCom), 2010 IEEE International Conference on*, pp. 162–170, IEEE, 2010.

[4] R. Chavarriaga, H. Bayati, and J. D. Millán, "Unsupervised adaptation for acceleration-based activity recognition: robustness to sensor displacement and rotation," *Personal and Ubiquitous Computing*, vol. 17, no. 3, pp. 479–490, 2013.

[5] A. Henpraserttae, S. Thiemjarus, and S. Marukatat, "Accurate activity recognition using a mobile phone regardless of device orientation and location," in *Body Sensor Networks (BSN), 2011 International Conference on*, pp. 41–46, IEEE, 2011.

[6] N. Kale, J. Lee, R. Lotfian, and R. Jafari, "Impact of sensor misplacement on dynamic time warping based human activity recognition using wearable computers," in *Proceedings of the conference on Wireless Health*, p. 7, ACM, 2012.

[7] K. Kunze and P. Lukowicz, "Dealing with sensor displacement in motion-based onbody activity recognition systems," in *Proceedings of the 10th international conference on Ubiquitous computing*, pp. 20–29, ACM, 2008.

[8] K. Forster, D. Roggen, and G. Troster, "Unsupervised classifier self-calibration through repeated context occurences: is there robustness against sensor displacement to gain?," in *Wearable Computers, 2009. ISWC'09. International Symposium on*, pp. 77–84, IEEE, 2009.

[9] K. Kunze, P. Lukowicz, H. Junker, and G. Tröster, "Where am i: Recognizing on-body positions of wearable sensors," in *Location-and Context-Awareness*, pp. 264–275, Springer, 2005.

[10] K. Kunze, P. Lukowicz, K. Partridge, and B. Begole, "Which way am i facing: Inferring horizontal device orientation from an accelerometer signal," in *Wearable Computers, 2009. ISWC'09. International Symposium on*, pp. 149–150, IEEE, 2009.

[11] S. O. Madgwick, A. J. Harrison, and R. Vaidyanathan, "Estimation of imu and marg orientation using a gradient descent algorithm," in *Rehabilitation Robotics (ICORR), 2011 IEEE International Conference on*, pp. 1–7, IEEE, 2011.

[12] J. J. Kuffner, "Effective sampling and distance metrics for 3d rigid body path planning," in *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, vol. 4, pp. 3993–3998, IEEE, 2004.

[13] D. Q. Huynh, "Metrics for 3d rotations: Comparison and analysis," *Journal of Mathematical Imaging and Vision*, vol. 35, no. 2, pp. 155–164, 2009