2018 International Conference on Indoor Positioning and Indoor Navigation (IPIN), 24-27 September 2018, Nantes, France

Supervised Learning for Yaw Orientation Estimation

Tobias Feigl*† tobias.feigl@fau.de
Christopher Mutschler†‡ christopher.mutschler@iis.fraunhofer.de
Michael Philippsen* michael.philippsen@fau.de

*Programming Systems Group
‡Machine Learning and Data Analytics Lab
Friedrich-Alexander University Erlangen-Nürnberg
Erlangen, Germany

†Machine Learning and Information Fusion Group
Fraunhofer Institute for Integrated Circuits IIS
Nuremberg, Germany

***Abstract*—With free movement and multi-user capabilities, there is demand to open up Virtual Reality (VR) for large spaces. However, the cost of accurate camera-based tracking grows with the size of the space and the number of users. No-pose (NP) tracking is cheaper, but so far it cannot accurately and stably estimate the yaw orientation of the user's head in the long-run.**

Our novel yaw orientation estimation combines a single inertial sensor located at the human's head with inaccurate positional tracking. We exploit that humans tend to walk in their viewing direction and that they also tolerate some orientation drift. We classify head and body motion and estimate heading drift to enable low-cost long-time stable head orientation in NP tracking on 100 m × 100 m. Our evaluation shows that we estimate heading reasonably well.

I. Introduction

VR drives innovation in applications for theme parks, museums, architecture, training, simulation, etc. They all can benefit from multi-user interaction, from areas beyond 20 m × 20 m, and from natural movement without motion sickness, but today's Simultaneous Localization and Mapping (SLAM) based pose estimation only achieves precise and drift-free tracking under restricting conditions (e.g., small rooms, static scenes/no or only a few moving objects, and homogeneous lightning) [1]. Moreover, today's state-of-the-art VR systems for small areas use camera-based motion tracking. Tracking accuracy decreases and cost grows strongly both with the camera resolution and the size of the area, and tracking more users needs more cameras to avoid occlusion.

Conceptually, no-pose (NP) tracking systems based on Ultra-wideband (UWB) that track single positions instead of the full pose (position and orientation) are cheaper and can work with larger tracking areas and more users. In contrast to camera-based tracking of the full pose, they are inaccurate and their positions cannot be combined to derive the pose. The absolute head orientation must thus be estimated separately.

Current low-cost Head-Mounted Display (HMD) units (with their inertial measurement units (IMU) such as accelerometers, gyroscopes, and their magnetometers) can be used to estimate head orientation even with lower latency and better immersion than camera-based systems. But, while it is possible to estimate the correct absolute pitch and roll orientation with accelerometer and gyroscope sensors, in practice an IMU-based estimation of the yaw orientation, i.e., of the rotation angle around the vertical body axis, is still inaccurate. First, magnetometers are unreliable in many indoor and magnetic environments and provide a wrong absolute yaw orientation [2]. Second, dead reckoning based on relative IMU data leads to drift and (after a while) to a wrong yaw orientation estimation [3]. Third, state-of-the-art filters fail to provide reliable motion direction estimates on noisy low-cost sensors as they require either noise-free accurate sensor models or military-grade sensors [4]. It is virtually impossible to tune the parameters of the Kalman filter, i.e., Kalman gain or process- and measurement-noise, so that they correctly describe the dynamics of sensor biases, human head motion, and the resulting non-linearities [5]. Even methods that stabilize a state-of-the-art Kalman filter based on known sensor states, e.g., shoe-mounted sensors [3] or context based hand-held sensors, fail to estimate the absolute yaw orientation of the head [6].

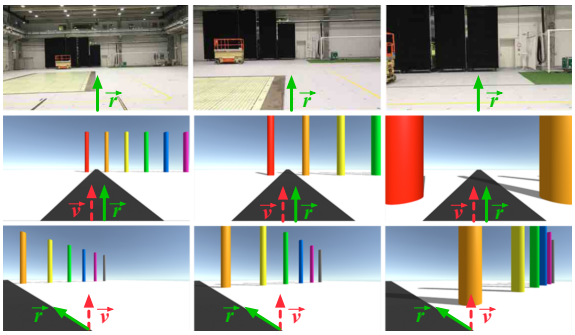
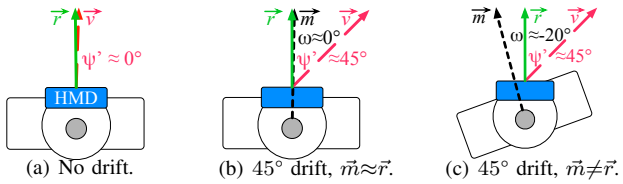


Fig. 1. Real (top) and virtual world (middle and bottom). Real head/view direction $\vec{r}$, virtual view direction $\vec{v}$. Middle: no drift. Bottom: sensor drift of 45° to the right. Although in the bottom row the user still looks and walks into the same view direction $\vec{r}$ as in the top and middle rows, his/her virtual view is drifted by 45° to the right. Hence, instead of virtually walking towards the blue pillar s/he walks sidewards and approaches the orange pillar in the VR. The user can either rotate the head by 45° or walk sidewards to adjust $\vec{r}$ to $\vec{v}$, or both. If the virtual view $\vec{v}$ diverges from the real view $\vec{r}$, a user is affected by motion sickness that grows with the offset between $\vec{v}$ and $\vec{r}$.

A wrong orientation estimation results in a mismatch of the real world and the VR display. The upper row in Fig. 1 shows the view of a user who walks straight ahead with his/her head oriented in the direction of the movement. In the VR (middle row of Fig. 1) this movement should lead through the clearance between the red and orange pillars. However, under drift (the bottom row shows a 45° yaw drift/offset) the same movement

978-1-5386-5635-8/18/$31.00 ©2018 IEEE



(a) No drift. (b) 45° drift, $\vec{m}\approx\vec{r}$. (c) 45° drift, $\vec{m}\neq\vec{r}$.

Fig. 2. Drift and its effect. Bird-eye's perspective onto a user wearing a HMD. Offset ψ' between real head direction $\vec{r}$ and virtual direction $\vec{v}$; offset ω between $\vec{r}$ and movement direction $\vec{m}$.

leads to a displacement from right to left as a wrong head/view direction $\vec{v}$ is used to render the VR images. For the user the direction of the movement does not fit to the VR view. This can cause motion sickness.

The key idea of this paper is to combine inaccurate positional tracking (horizontal error of $\pm 20\ cm$) with relative IMU data to achieve a long-time stable yaw orientation while the user is (and keeps) walking naturally and also freely rotating his/her head. Under the assumption that humans mostly walk in their viewing direction we extract features from sensor signals, classify the relation between real movement direction and real head orientation (with supervised machine learning), and combine this with absolute tracking information. This yields an estimation of the absolute yaw orientation.

The rest of this paper is structured as follows. Sec. II describes the problem. Sec. III describes our head orientation estimation including a signal processing and feature extraction pipeline for head-mounted IMUs that allow for head-related movement classification. Sec. IV evaluates technical aspects of our work. Sec. V reviews related work before we conclude.

II. Problem Description

Sensors for relative movement estimation drift and inevitably cause a wrong yaw orientation in the long-run. But often relative sensors are the only option as absolute sensors (such as magnetometers) do not work reliably in practice.[1]

Fig. 2 illustrates different drift scenarios from a bird-eye's perspective onto a user with an HMD (in blue). In Fig. 2(a) there is almost no drift ($\psi'\approx 0°$), see also middle row in Fig. 1. The user's real head direction $\vec{r}$ is close to his/her virtual head direction $\vec{v}$, i.e., the angle between $\vec{r}$ and $\vec{v}$ is zero. Movements feel natural as the VR image is rendered with the correct head-to-body pose, i.e., with a correct absolute yaw orientation. In Fig. 2(b) the angle between $\vec{v}$ and $\vec{r}$ differs by $\psi'\approx 45°$, see also bottom row in Fig. 1. When the user moves in the direction of $\vec{m}$ s/he recognizes this as unnatural translation of the rendered image towards $\vec{v}$. Unfortunately, with an unknown head direction $\vec{r}$, a VR system does not know ψ' and hence cannot align $\vec{v}$ closer to $\vec{r}$.

To understand our approach let us simplify first. Assume that a fine-grained absolute position tracking (e.g., a radio-frequency-based UWB system) of users is available, both with respect to coordinates and time stamps. From a user's absolute positions over time we can then extract a trajectory vector $\vec{m}$ from pairs of consecutive absolute positions. With the assumption that users always look forward in forward movements, i.e., $\vec{m}$=$\vec{r}$, a VR system *can* then deduce $\vec{r}$, adjust $\vec{v}$, and eliminate the drift that causes motion sickness. Of course in reality the head is not always aligned, see Fig. 2(c). If the user looks to the right by ω=$-20°$ the adjustment of $\vec{v}$ as described above would still yield a drift of 25°. The same happens if $\vec{m}$ differs from the user's real trajectory.

Instead of directly estimating the yaw orientation from IMU sensor data, we solve the problem of how to use that data to detect moments in which the user moves forward into his/her viewing direction, because for those $\vec{m}$=$\vec{r}$-moments or ω=0°-moments, we know how to adjust the drift.

[1] With accelerometers one can absolutely estimate pitch and roll, but both gyroscopes and accelerometers cannot absolutely estimate the yaw orientation.

III. Estimation Of The Human Head Orientation

With supervised machine learning we classify ranges of ω. If among all the ranges the ω=0°-moment class has the highest probability, we have detected an $\vec{m}$=$\vec{r}$-moment. From the IMU data (accelerometer) we extract the linear acceleration component, i.e., the real translational movement (without gravity) in every direction, and use it in combination with the gyroscope data to derive specific features that characterize and represent a certain range of ω. We train a classifier for all the ω-classes a-priori on pre-recorded and labeled feature data. At runtime, we classify ω on live sensor data to detect a ω=0°-moment if this class yields the highest confidence.

Fig. 3 outlines our processing pipeline. First, we smooth the raw sensor signals with digital filters (Sec. III-A). In a training step we extract features (Sec. III-C) for known ranges of ω on labeled training samples to train the classifier. While a fine-grained resolution of ω-range classes improves the classification and its confidence, it also needs more data for the training and more CPU cycles for the classification (Sec. IV-B suggests parameters). At runtime, the trained classifier processes the features of (smoothed) unknown signals and returns the best-fitting ω-range class and its classification confidence. In ω=0°-moments we determine the head orientation drift ψ'.

A. Signal Processing

Raw accelerometer and gyroscope data from a low-cost IMU sensor are too noisy to extract reliable features. They need pre-processing. Typical low-cost accelerometers acc^{raw} track gravity and acceleration at 200 Hz up to $\pm 16\ g$. Gyroscopes track the raw angular velocity gyr^{raw} at 200 Hz up to $\pm 2000\ °/s$. To describe the user's head-to-body-pose accurately, we need to analyze the head's motion, its pose, and its rotation. Therefore, we separate the acc^{raw} into its gravity component acc_{grav} (which describes the pose, i.e., pitch and

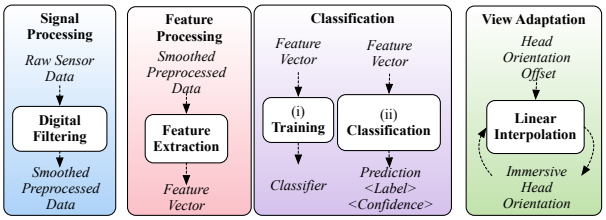
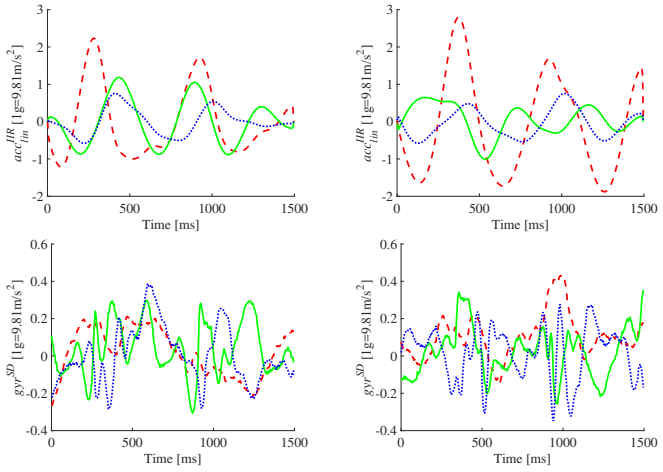


Fig. 3. Head orientation estimation processing pipeline.



(a) ω=0°. (b) ω=45°.

Fig. 4. Top: IIR (LP,HP)-filtered linear accelerations data acc_{lin}^{IIR}. Bottom: SG-filtered gyroscope data gyr^{SG}. About one gait cycle, (=2 foot steps) shown. Directions: red dashed lines X or ψ (yaw), green solid lines Y or θ (pitch), and blue dotted lines Z or ϕ (roll).

roll) and the linear acceleration acc_{lin} (which describes the motion).

We also filter both the gyr^{raw} and acc^{raw} data to smooth it. The details of the input data processing follow.

1) IMU Data Preprocessing: For the pre-processing of the IMU data we use sliding windows, i.e., six windows to store the data of acc^{raw} (3 axes) and gyr^{raw} (3 axes). In contrast to a simple moving average filter, our Savitzky-Golay-filter (SG) removes signal noise, vibrations, and (small) motion artifacts while significant changes remain in the filtered signal. A small window of length n=25, a polynomial order of P=3 (higher orders capture noise only), and the usual SG-filter's convolution coefficients C_k are sufficient.

2) Accelerometer Data Filtering: To pre-process the gyr^{raw} data we eliminate noise with our SG-filter into gyr^{SG}. However, as the gravity component of the acceleration signal reflects the real pose (pitch and roll) and the linear acceleration reflects the motion, we separate the accelerometer signal in its gravity and linear components: we first isolate the linear acceleration with a linear recursive filter and then we derive the gravity (gravity = raw acceleration minus linear acceleration). The upper graphs in Fig. 4 show acc_{lin}^{IIR}, the filtered signals that only hold the linear acceleration components after the application of a high-pass (HP) filter (that removes low-frequent gravity components) and a low-pass (LP) filter (that purges high-frequent noise and motion artifacts).

In preliminary tests we compared FIR (finite impulse response) and IIR (infinite impulse response) filters. We found that in our case, IIR filters outperform FIR filters as they yield the smallest error and the fastest runtime with the smallest delay (at a similar filter order). For each accelerometer axis we thus use two IIR-filters (LP/HP) with a fast and reliable Butterworth filter design. The linear and gravity acceleration components are not SG-filtered but only IIR filtered.

As human motion happens below a frequency of 18 *Hz* [7] we sample the signal above 40 *Hz* (in line with the Shannon-Nyquist theorem). The LP filter compensates for artifacts such as extremely fast head movements or vibrations. Some results from our experiments: (1) Frequencies in $[5; 20]$ *Hz* cover all human motions in VR. (2) Users move/walk even slower in VR. We set the cut-off frequency of the HP filter to 5 *Hz* to compensate for the long-term vertical signal drift.

Each accelerometer measurement x_i is pushed into a sliding window of size n=12,000 that can hold the data of one minute (200 measurements per second). This gives the IIR filter enough history and allows to capture all available frequencies/activities that are embedded in the signal. The Butterworth filter gives the filtered accelerations y_i for the x_i as $y_i = \frac{1}{a_0}(\sum_{i=0}^{N} b_i \cdot x_{i-1} - \sum_{j=1}^{M} a_j \cdot y_{i-j})$, where $a_{0\cdots M}$ and $b_{0\cdots N}$ are the filter coefficients of the feedback and the feed-forward. It is an engineering task to derive optimal filter orders (LP: N_{low}=M_{low}=3; HP: N_{high}=M_{high}=1). We found that these values are a reasonable trade-off between filter depth and computation time.

As the decomposition of the acceleration signal (into acc_{grav}^{IIR} and acc_{lin}^{IIR}) is tied to the filtering and thus to the loss of information of unique sensor characteristics we also use the SG-filtered raw acceleration (acc^{SG}) to preserve both the signal characteristics and the relation between the gravity and the linear acceleration components.

This yields a total of 4 streams of pre-processed data (with 3 axes each) that we use to extract features: SG-filtered gyroscope rotations gyr^{SG}, SG-filtered accelerations (includes pose and motion) acc^{SG}, and two IIR-filtered accelerations, separated into a stream for the gravity (pose) acc_{grav}^{IIR} and one for the linear (motion) accelerations acc_{lin}^{IIR}.

B. Influence of Motion IMU Signals

We are now ready to extract information on the head-to-body-pose from the pre-processed data. To simplify the explanation let us assume two (classes of) ω-angles, namely ω=0° and ω=+45°. (Sec. IV evaluates a classifier for 7 classes of ω-angles.) Fig. 4 shows the IIR-filtered linear accelerations acc_{lin}^{IIR} in its upper row and the SG-filtered gyroscope signals gyr^{SG} in its bottom row. On the left the user walks with the head looking forward. On the right the yaw orientation is ω=+45°. We can clearly identify oscillations in the X-axis (red dashed curve) as the user's head moves up and down with each step. Since the head also moves back and forth (Z-axis) and sidewards (Y-axis) with every step, there are smaller oscillations in the blue and green curves. As the upper graphs look alike, a classifier that detects whether a user walks straight ahead into the direction of view or whether the head is turned to the side, barely works with the acceleration signals alone.

When users walk, they balance out their heads' up and down movement with a nodding motion in the pitch-axis θ (green solid line). In the ω=0°-moment (Fig. 4(a)) the nodding results in a prominent oscillation of θ. If the head is turned sidewards with respect to the movement direction then the nodding is distributed over both the ϕ (blue dotted line) and θ (green solid line) axes. There are similar rotations to balance out the other head movements that can be seen in the acceleration peaks.

Hence, the gyroscope data is also necessary to classify the ω-angle. While in forward movements the acceleration signal spreads its forces (both gravity and linear) over all axes, the forces mostly concentrates in the Z-axis in the ω=0°-moment and shifts to the Y-axis when the head is turned sidewards. Thus, the correlation between the Y- and Z-axes gives some insight into how far the head is turned to the left/right.

Besides the linear acceleration that represents the movement of the head we are also interested in the pose of the head as it indicates how humans hold their heads. As gravity indicates the pitch and roll of a sensor with respect to the ground, the pose of the head is based on the gravity components per axis. Intuitively, that is why we also need the IIR-filtered gravity stream acc_{grav}^{IIR} that is left out from Fig. 4.

C. Feature Selection

Now that we have motivated that the 4·3=12 signal streams hold enough information to extract and classify the ω-angle, we turn them into a stream of features that capture the intuition and that the classification algorithm can make use of.

Potential features from the literature [8] have different runtimes (both during training and runtime) and yield different confidences. It is an engineering task to find a minimal set of features that uniquely describe a class and that are separable from the features of other classes. Recall that for the signal filtering in Sec. III-A we picked the window size according to the needs of the filters. Similarly, the computations of the features have to process a certain window of the (filtered) signal data. The above discussion of Fig. 4 has motivated that at least the signals of one full gait cycle are needed to classify the ω-angle. From a pre-examination we know that adding more gait cycles does not improve the results but slows down the feature extraction. When users walk at a normal speed in a typical VR setup, one cycle fits into about 1000 ms, i.e., into 200 (filtered) signal values.[2] We found that for each of the 12 signal streams the following four features (computed over the full window of size n=200) keep the required CPU resources low while yielding a good classification. We give some rationale why they capture what the classifier needs.

1) If a sliding window would hold the sensor data of *exactly* one gait cycle, the **mean** μ of the values remains constant, even if the window slides over the motion. Conceptually, there are always one or two full waves in the window (depending on the axis), only the cut-off point varies at which the window starts. If the sliding window holds more (or less) than a full gait cycle, then the mean value varies with the cut-off point, as a varying fragment of a wave is an extra (or missing) part of the window. μ then oscillates and captures at which spot of the gait cycles the sliding window starts (at any point of time). μ also helps classifying the moments when the user is standing (ω =?-moments) since when the user is not moving, μ no longer oscillates (that much) and stays around 0.

[2]According to [9] users perform 1.5 $steps/s$ when walking at 1.4 m/s in reality but tend to walk slower in VR: slow speed = 0.75 m/s, normal speed = 1.0 m/s, and fast speed = 1.25 m/s.

2) The **standard deviation** SD represents the intensity of signal fluctuations (due to nodding, balancing, etc.). This is a reasonable choice for our classification task as the head turns right/left (Y-axis) in consequence of the walking and as this movement (and thus its SD) is stronger in ω=0°-moments than when ω is at $\pm$45°. Above we discussed that the gyroscope data is necessary to distinguish positive from negative ω-values; thus we also need the SD of gyroscope data.

3) As we have argued above, the **correlation between the Y- and Z-axes** $corr_{yz}$ gives insight into how far the head is turned/rotated around the yaw axis. The X-axis values are more or less irrelevant. In ω=0°-moments the force is mainly present on the Z-axis whereas in ω=$\pm$45°-moments it is spread over the Y- and the Z-axes. For use as a feature, we therefore calculate the correlation $\kappa = \tan^{-1}(\frac{Z}{Y})$. Due to the orientation of the coordinate systems, κ is in $[0°, 180°]$ as users cannot turn their heads to their backs. The classifier can detect a ω=0°-moment when κ=90°, with a tolerance of $\pm$20° as most humans do not notice a yaw drift below 20° [10].

4) We calculate a **Principal Component Analysis** pca and use strong eigenvectors to map the data window onto a single value that we use as a fourth feature. All the eigenvectors together describe the time-dependent variances between all the values that exist within the window, i.e., they describe how the signal fluctuates. For instance, for the ω=+45°-moments from Fig. 4(a) there are eigenvectors that describe the slight kink in the X-value of the acceleration at around 600 ms in relation to the peak at around 900 ms.

We consider the sliding window to be a 200-dimensional vector on which we apply a singular value decomposition (SVD) to derive 200 eigenvectors and eigenvalues that describe how the values are distributed in the window. As our signals are continuous this yields windows that contain similar eigenvectors (with similar eigenvalues). For each window we create a histogram over all the eigenvectors (multiplied with the data vector and scaled by the eigenvalue to get a single value) and let a binning algorithm find meaningful clusters of eigenvectors. We then use the bin with the largest value (the highest variance/information density) as the pca feature.

We found that the set of 12 (one per input stream) strongest histogram peaks provides a good basis to separate ω=?- and ω=0°-moments from other ω-angles. Moreover, our pca can also distinguish negative from positive ω-values as both the gyroscope and the accelerometer data have significant signal variations that uniquely describe those movements.

From the 4·3 signal streams we thus extract a total of 4·3·3=36 streams (μ, SD, and pca), plus 4 streams for $corr_{yz}$ to combine the axes. In Sec. IV-B we evaluate the runtime with respect to the importance to the 36+4=40 feature streams and show that in combination they separate the ω-angles well enough to detect the ω=0°-moments that are needed to reduce the drift of the VR-display. Other research also uses similar sets of features. They are a good trade-off between model complexity, computation time, and classification rate [11].

IV. Evaluation

After sketching our measurement setup we evaluate the accuracy of ω-moment detection and justify our feature selection.

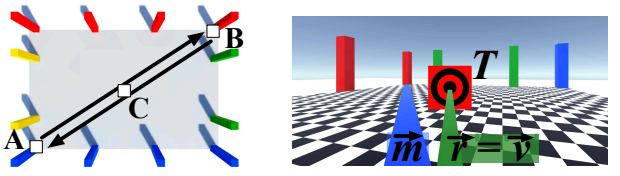
A. Measurement Infrastructure and Setup

On a tracking space of about 45 m × 35 m, all experiments use a Samsung Galaxy Note 4 smartphone (Android 6.0.1, Qualcomm Snapdragon 805 CPU, 3 GB RAM) attached to a Samsung Gear VR HMD (SM-R320 equipped with a BOSCH 6 DOF IMU sensor BNO-055). The IMU measures accelerations within $[\pm 2, \pm 16]$ g and gyroscope rotations within $[\pm 250, \pm 2000]$ $^\circ/s$. The Android sensor API cuts off sensor readings above 200 *Hz*. Although a sampling rate of 40 *Hz* would suffice for the frequency range of human motion [7] and satisfy the Nyquist-Shannon theorem, we still use 200 *Hz* since the OS and JVM cause unpredictable power state switches and some timing jitter. The 200 *Hz* sampling rate also stabilizes the IIR-filters. The filter parameters and assumptions that we made in Sec. III are in line with the specification and limitations of the hard- and software we use. In addition to the IMU data, for the training and the evaluation of the classifiers in Sec. IV-B we also need highly precise yaw orientation measurements to label the IMU data. We obtain such labels by means of a Nikon iGPS system, an optical laser-based tracking system with an average vertical/horizontal accuracy below 10 mm at 20 *Hz*. We use a head-mounted apparatus that carries two locatable objects at a distance of 50 cm from which we calculate the absolute yaw orientation. We also use the iGPS system to (re)calibrate the yaw orientation to measure the drift before/after our studies.

Fig. 5(a) is a bird's-eye view onto our VR measurement scenario which is computationally efficient so that we achieve a constant frame rate of 60 *Hz*. Fig. 5(b) is the corresponding ego-perspective with the visuals that guide the participants of our experiments. The blue line $\vec{m}$ on the ground always leads to the red pillar (B) or to the blue pillar (A). The target T and a green line from the user towards T move with the user and help retain a desired ω-angle of the head. The target T is placed according to the ω-angle used in an experiment. Users are asked to walk naturally but to always aim their head's yaw at T with an arbitrary pitch and roll, using the green line that is always in their middle of the screens.

B. Classification Performance

To evaluate the feature selection from Sec. III-B and the classifier performance we collected data with a group of 34 subjects (avg. age 23.16 [18, 36] years; avg. height 1.74 [1.49 - 1.81] m; 20 male, 14 female; nobody disabled or handicapped) and let them walk 10 times naturally from A to B on path $\vec{m}$ and back, see Fig. 5. We introduced the participants to the setup and to the purpose of the measurements beforehand. While we collected the IMU data from the HMD we also measured $\vec{r}$ with the iGPS system to obtain precise yaw orientations, i.e., a labeling of the IMU measurements. We asked the users to walk naturally and relaxed at a normal speed (avg. 0.87 m/s, min.: 0.58 m/s, max.: 1.19 m/s, SD: 0.18 m/s) and to keep their heads at a fixed ω-angle $[-45°; -30°; -15°; 0°; +15°; +30°; +45°]$ with the help of the target T and the green line. Our VR system also used a voice feedback to alert a user when s/he undercut or exceeded the target-ω by more than 5°. To obtain a close-to-natural head pose while walking, we did not enforce a rigid head pitch and roll. The collected data shows the typical compensation movements [10], [12] of the heads along the users' trajectories (Sec. III) and the data also includes natural noise and jitter around the target-ω. In a post-processing step we cleaned the data both from the moments when ω was outside of the target zone and from the moments when the users turned around at B or A. We recorded about 8 h of movement data. According to Sec. III-B we pre-processed it and extracted the feature streams. To avoid minority oversampling, we removed some (random) samples so that all ω-classes have the same size.



(a) Users walk 50 m between A and B or stand at C.

(b) The target T helps retain a desired ω-angle. Blue line = walking path, green line = view direction.

Fig. 5. Top-view (a) and first-person-view (b) of our VR scenario.

1) Classifier Comparison: We study the performance of three classifiers. First, a **Support Vector Machine (SVM)** with a cubic kernel function $K(x_q, x_i) = (1 + \gamma \cdot x_q^T x_i)^d$. Since we need a multi-class classification we use a One-vs-All SVM. Second, a Classification And Regression Tree (CART) **Decision Tree (DT)** model. According to preliminary experiments, a DT performs best for the ω-classification when

TABLE I
Results of a 10-fold cross-validation in % for 3 classifiers: SVM classifier in the upper numbers of the cells, DT in the middle, k-NN at the bottom. True ω-classes along the vertical axis, classification results horizontally. Correct classifications in **bold** and misclassifications in normal font. Shaded: success rate = correct classifications of ω=0°-moments.

	ω=-45°	ω=-30°	ω=-15°	ω=0°	ω=+15°	ω=+30°	ω=+45°	ω=?	
SVM	**83**	14	0	0	0	0	0	0	ω=-45°
DT	**78**	15	1	4	1	1	1	0	
k-NN	**66**	26	1	4	2	1	1	0	
	9	**80**	5	4	1	1	0	0	ω=-30°
	13	**67**	8	7	4	2	1	0	
	10	**67**	13	5	4	1	0	0	
	1	4	**77**	3	11	3	1	0	ω=-15°
	1	7	**60**	5	19	8	1	0	
	1	5	**64**	3	24	3	0	1	
	1	4	4	**86**	4	1	0	0	ω=0°
	2	7	7	**72**	10	3	0	0	
	1	7	9	**71**	11	0	0	0	
	1	1	10	6	**74**	9	0	0	ω=+15°
	0	1	13	10	**61**	16	1	0	
	0	0	11	9	**62**	8	0	1	
	0	2	3	2	10	**76**	7	0	ω=+30°
	1	1	6	3	12	**67**	10	0	
	0	1	3	3	23	**65**	5	1	
	1	1	0	0	1	11	**86**	0	ω=+45°
	1	1	1	1	1	16	**79**	0	
	0	1	1	1	2	21	**74**	0	
	1	0	0	0	1	0	0	**99**	ω=?
	0	0	0	0	1	1	0	**99**	
	0	1	0	0	0	0	0	**99**	
	ω=-45°	**ω=-30°**	**ω=-15°**	**ω=0°**	**ω=+15°**	**ω=+30°**	**ω=+45°**	**ω=?**	*Truth*

Classification/Prediction

configured with no more than 100 splits, a minimal leaf number of 1, the Gini diversity index I_G as split criterion, and a subsequent pruning that keeps more than 10 parents per leaf. Third, a cubic **k-Nearest Neighbor (KNN)** that according to preliminary experiments works best for us when it is used with a distance parameter k=3, a distance function $(X\times Y)^n \rightarrow (X \rightarrow Y)$, and cubic Minkowski distance metrics.

With each of these classifiers we run a 10-fold cross-validation against the 4·3·3+4=40 labeled feature streams. It splits the data into 10 equally large sub-samples and uses 9 sub-samples for the training of the classifier and the remaining sub-sample for the validation. This is repeated 10 times with each sub-sample once being the validation set. All classifiers almost perfectly detect ω=?-moments. When the participants are walking, the SVM classifier yields the highest correct classification rate, i.e., the highest confidence for all ω-angles (min: ω=+15° at 74%, max: ω=? at 99%). All classifiers clearly separate ω=0°-moments from other ω-ranges. The SVM classifier correctly classifies most (**86**%) of them.

To process the 40 features of all streams, the SVM classifier takes 259 µs (per update) whereas DT and k-NN take 29 µs and 1091 µs, resp. As the SVM classifier outperforms the DT classifier, the extra runtime is worth it.

2) Feature Comparison: We train and use the classifiers both with different subsets of the features (μ, SD, $corr_{yz}$, and pca) and on different subsets of the input streams (acc^{raw}, gyr^{raw}, acc^{SG}, gyr^{SG}, acc^{IIR}_{grav}, and acc^{IIR}_{lin}). No matter for which combination of features and streams we use the three classifiers, SVM clearly outperforms the other two. Thus below we discuss the numbers of the SVM classifier only. The more features and input streams are used, the better the success rate gets (up to a maximum of 86%). The more features and the more data there are to process, the longer a single classification takes (up to 259 µs).

When only the raw signals are used to classify ω-moments the best achievable success rate is 78%. It is better to use SG-filtered streams instead (81%). The sensor-specific information (noise, response time, min-max-range) that is lost by the filtering does not hurt (SD on acc^{raw} 69% vs. acc^{SG} 63%) as the combination of several features and filtered input streams outperforms the classification based on raw data streams (even in all possible combinations).

Let us now discuss the effect of using an individual feature. The pca feature on its own provides the highest success rate (71%). The $corr_{yz}$ feature extracts similar information from the streams and closely follows at a similar computational cost. The features μ and SD perform worse (regardless of how many input streams are used). This is because μ only describes the cut-off points of the gait cycle in the sliding window, and SD only provides energy information, i.e., it only separates ω=? from ω=0° as their variances differ the most. More input streams only repeat this information and thus do not increase the success rate.

Combining the features yields higher success rates. While a combination of μ and SD already boosts the classification (78%), an extra $corr_{yz}$ (83%) or pca (83%) helps even more as these features describe different characteristics, e.g., motion direction or an abstract signal pattern. That is also why combining $corr_{yz}$ and pca alone performs worse (76%): the movement's state and type are missing. The complete feature set on all input streams yields a success rate of 86%.

3) Resulting Classification Accuracy: For the purpose of the drift elimination the classification results are even better as humans tolerate a drift of 20° without noticing it [10]. Misclassifications to an adjacent ω-class ($\pm\omega$=15°) are also tolerable. Then the SVM-classifier yields a correct result in 86+3+6=**95**% of the cases.

After this coarser classification we compare consecutive results with the gyroscope signals. Since there must be a correspondence as humans cannot turn their heads too much between samples we can fix ***all*** remaining misclassifications.

TABLE II

SUCCESS RATES (= CORRECT CLASSIFICATIONS OF ω=0°-MOMENTS) OF THE SVM CLASSIFIER IN % WITH A SUBSET OF THE FEATURES (VERTICAL) AND ON A SUBSET OF FILTERED INPUT DATA STREAMS (HORIZONTAL). SYMBOL – INDICATES A FEATURE OR DATA STREAM THAT IS LEFT OUT. THE **BOLD 86** IS ALSO IN TABLE I. UNDERLINED NUMBERS ARE DISCUSSED.

Features				Streams: acc^{raw}	– | gyr^{raw}	acc^{raw} | gyr^{raw}	–	acc^{SG} | –	– | gyr^{SG}	acc^{SG} | gyr^{SG}	– | –	acc^{IIR}_{grav}	acc^{SG} | gyr^{SG} | acc^{IIR}_{grav}	– | acc^{IIR}_{lin}	acc^{SG} | gyr^{SG} | acc^{IIR}_{lin}	acc^{SG} | gyr^{SG} | acc^{IIR}_{grav} | acc^{IIR}_{lin}	CPU (all streams) [µs]
μ	–	–	–	66	49	6	66	47	68	34	61	61	62	62	61
–	SD	–	–	69	60	68	63	75	64	69	57	56	58	59	60
–	–	$corr_{yz}$	–	14	6	67	14	14	6	9	65	73	69	68	62
–	–	–	pca	71	12	66	64	12	65	11	64	68	70	71	63
μ	SD	–	–	66	54	60	67	39	67	40	77	78	77	78	93
μ	SD	$corr_{yz}$	–	73	50	76	71	42	70	40	79	83	83	83	109
μ	SD	–	pca	73	56	77	73	44	72	44	79	80	81	83	161
–	–	$corr_{yz}$	pca	70	13	74	70	25	68	19	72	73	75	76	87
μ	SD	$corr_{yz}$	pca	73	52	78	73	42	72	41	81	84	83	**86**	259

C. Applicability For Real-World Use Cases

Being able to detect ω=0°-moments and to purge the drift is of course only relevant in practice if ω=0°-moments do occur often enough. To check how often and for how long humans walk towards their viewing direction in a freely walkable, large-scale and multi-user VR, we asked 79 (other) subjects (avg. age 32.34 [18, 63] years; avg. height 1.71 [1.49 - 1.96] m; 43 male, 36 female; nobody disabled or handicapped) to explore a virtual museum [13] that holds six different exhibits of real sized dinosaurs on 45 $m \times 35$ m.

We used the same Samsung GearVR HMDs to record the sensor streams (acc_{raw} and gyr_{raw}) and our UWB tracking system (CEP_{95}=20 cm at 20 Hz) that tracks the absolute positions of the heads (similar to [14]). We use the iGPS apparatus to (re)calibrate and measure the absolute head orientations before and after each walk.

On average, each participant performs 41 separate ω=0°-moments within a 3 minute interval (min: 36, max: 54, SD: 7.4) at an average walking speed of 0.74 m/s (min: 0.56 m/s, max: 1.60 m/s, SD: 0.14 m/s) and an average distance of 1.18 m (min: 0.51 m, max: 2.27 m, SD: 0.37 m). Hence our

calibration can kick in every 4.4 s (=180 s / 41). This shows that users tend to walk more frequently and longer towards their viewing direction, i.e., there are enough ω=0°-moments for a robust yaw estimation.

D. Possible Limitations

It is not a limitation in practice that our approach relies on the availability of the sensor data of a full gait cycle in a fixed-size sliding window and also of an (inaccurate) absolute position vector. The reasons are: (1) Although users walk slower in VR than in the real world, a sliding window of 1 s (200 *Hz*) suffices as it holds two steps and spans a position vector of about 1 m at a negligible positional error. (2) The evaluation in Sec. IV-C shows that in a real-world use case variable movement speeds do not impede the classification accuracy. (3) While a window length of 200 *Hz* is large enough for both walking and running, for higher velocities a larger window can be used. (4) For relevant window sizes, both the feature extraction and the classification are fast enough to be hidden in other latencies. Both the SG- and IIR-filters yield a signal stream that is delayed by 47 ms on average with respect to the absolute position stream (min.: 27 ms, max.: 76 ms, SD: 6.3 ms, in theory 1/8th of the sliding window size). We found that the pre-processing of the radio-frequency based position tracking system suffers from an 'over-the-air' transmission delay and is also late by 57 ms on average (min.: 32 ms; max.: 76 ms, SD: 5.9 ms). Thus the total delay/shift averages out and is hence negligible. (5) A pre-processing step helps to only run the classification when the users walk on straight trajectories (minimal distance covered is $\geq$1 m or the duration is $\geq$1 s).

The accuracy of the training data poses a threat to the classification performance and hence to the effectiveness of our technique. The classification can only be as accurate as the data it is trained on. The more diverse the training data is (e.g., more participants, more variations and different types of motions), the higher is the confidence of the classification at runtime. But even without such a (re)training, we already see our method working fine in daily use, even for unknown motion of new VR users.

It is not a limitation of our approach that we only estimate the yaw orientation on ω=0°-moments. We simply do not need a full 3D pose (re)calibration, since pitch and roll are already accurate from accelerometer and gyroscope data.

V. Related Work

VR systems need accurate head orientations. Traditionally, they use (local) reference systems with absolute orientations of the tracked objects. The head pose is often estimated in the tracking system instead of near the HMD. To reduce latency and to increase update rates we do the latter. Foxlin [15] also uses the IMU so that the reference system only needs to stabilize the estimations over time. But in contrast to our users, his users cannot move freely in the limited tracking space. We also do not need to stabilize a filter with 'ground truth moments' from feet-mounted sensors when a foot hits the ground to estimate the absolute orientation [3]. Moreover, the feet (or torso) orientation does not tell the head orientation since these are different rotation systems. Until today, there is no publicly known noise measurement model that accurately represents the dynamics of the head and that reliably works for longer than a minute [4].

Kinect, Tango, and ARKit use SLAM that only works well under restricted conditions (small rooms, heterogeneous surface textures, static scenes, and homogeneous lightning). Because of methodical (labile feature detection [1]) and physical (depth sensing) limitations of RGB [1] and RGB-D sensors, today's SLAM cannot provide our dynamic and flexible VR experiences for immersive, freely walkable VR applications [16].

Orientation estimation techniques for NP tracking with IMUs differ from typical VR systems [17]. When highly precise and more expensive sensor data is available some works estimate both position and orientation with Bayesian filters [18]. The commonly used (extended) Kalman filters doubly integrate over the sensor signals and require precise models of the accelerometer sensor to properly extract linear acceleration [19]. For low-cost IMUs that lack precise measurement models Bayesian filters are not a viable solution, except when reliable magnetometer data can be exploited to reduce the yaw orientation error [20]. We do not need magnetometers.

While Human Activity Recognition can detect activities such as standing, walking, and sleeping with IMUs placed on body parts other than the head, the estimation of the head orientation is not the focus of those approaches. Windau and Itti [21] classify activities with head-mounted IMUs, but for good results a user's head must point into the direction of the movement. In contrast, our approach only needs a head-mounted IMU and users can freely move their heads. Beauregard et al. [22] use a helmet-mounted IMU, but for a different purpose, namely to estimate step length and heading. However, their neural network approach only works for fixed helmet-to-body orientations, i.e., as long as the orientations of the sensors align with the orientation of the body [23]. In contrast, our users can rotate their heads freely. Steed and Julier [6] compensate the yaw orientation drift of two different rotation systems (torso and hand) by merging both rotation systems so that the total relative yaw drift accumulates only along one axis. Their method thus cannot provide a correct absolute yaw orientation. They also exploit hand-to-body movement to derive the absolute yaw orientation from the context, e.g., they assume that VR users look at their hands when opening a door. In addition to the necessity of carrying sensor devices in the hands, this is a rather strong assumption.

Human-centric navigation systems that use Pedestrian Dead Reckoning with IMUs also need the orientation of sensors for reliable results. There are studies for various sensor combinations [5] and with different positions of the IMU on the body (hands [24], wrists, feet and legs [25]). Conceptually, PDR estimates both displacement (from step-detection and step-length estimation) and head orientation (from gyroscopes

and magnetometers) [4]. With feet-mounted sensors the resulting positional error can get as low as $[0.3\%; 2\%]$ of the total traveled real distance [2] since the systems can exploit recurring points when the IMU does not move. With solely head-mounted IMUs we cannot use this idea.

Some PDR research uses magnetometers to stabilize the head orientation estimates [26]. The idea is that each footstep leads to a specific head rotation. As the head rotation signal pattern can be learned a-priori, it can later be detected and used at runtime to correct head orientation errors [27]. Conceptually those approaches can also be used for other types of head-mounted IMU sensors. But there are more disadvantages that our approach avoids. First, such filters need to be parameterized (trained) *per user*. Second, if a user's body and head rotate in sync, such filters are confused and must be recovered manually. In contrast, our approach not only avoids unreliable magnetometers, we also avoid user-specific filters and are immune to synchronous movements of body parts.

VI. Conclusion

This paper shows how to estimate long-term stable absolute head-to-body orientations from inaccurate positions and noisy inertial sensors mounted at the head.

To achieve this goal, we presented a set of features to be extracted from filtered sensor data that (after some training with labeled data) a Support Vector Machine (SVM) classifier can use to reliably detect exactly those moments in which users walk with their heads facing forward and in which our VR system can thus derive the accumulated drift.

In typical multi-user and large-scale VR scenarios, e.g., museums and theme-parks, our technique can easily determine a user's head-to-body pose several times per minute, whenever the user looks in the direction of the movement, even with natural and relaxed motion (including the head).

Acknowledgments

This work was supported by the Bavarian Ministry for Economic Affairs, Infrastructure, Transport and Technology and the Embedded Systems Initiative (ESI).

References

[1] P. Meier, S. Ben Himane, S. Misslinger, and B. Blachnitzky, "Methods and systems for determining the pose of a camera with respect to at least one object of a real environment," 2015. US Patent App. 14/633,386.

[2] J. Bird and D. Arden, "Indoor navigation with foot-mounted strapdown inertial navigation and magnetic sensors," *IEEE Wireless Comm.*, vol. 18, no. 2, pp. 28–35, 2011.

[3] E. Foxlin, "Pedestrian tracking with shoe-mounted inertial sensors," *IEEE Compu. Graphics and Appl.*, vol. 25, no. 6, pp. 38–46, 2005.

[4] R. Harle, "A survey of indoor inertial positioning systems for pedestrians," *IEEE Comm. Surveys & Tut.*, vol. 15, no. 3, pp. 1281–1293, 2013.

[5] L. Zwirello, X. Li, T. Zwick, C. Ascher, S. Werling, and G. Trommer, "Sensor data fusion in UWB-supported inertial navigation systems for indoor navigation," in *Proc. IEEE Intl. Conf. Robotics and Automation*, (Karlsruhe, Germany), pp. 3154–3159, 2013.

[6] A. Steed and S. Julier, "Behaviour-aware sensor fusion: Continuously inferring the alignment of coordinate systems from user behaviour," in *IEEE Intl. Symp. Mixed and Augmented Reality (ISMAR)*, (Adelaide, Australia), pp. 163–172, 2013.

[7] D. Karantonis, M. Narayanan, M. Mathie, N. Lovell, and B. Celler, "Implementation of a real-time human movement classifier using a triaxial accelerometer for ambulatory monitoring," *IEEE Trans. Information Techno. in Biomedicine*, vol. 10, no. 1, pp. 156–167, 2006.

[8] N. Ravi, N. Dandekar, P. Mysore, and M. Littman, "Activity recognition from accelerometer data," in *Proc. 17th Intl. Conf. Innovative Appl. of Artif. Intelligence*, (Pittsburgh, PA), pp. 1541–1546, 2005.

[9] C. Abdelkader, R. Cutler, and L. Davis, "Stride and cadence as a biometric in automatic person identification and verification," in *Proc. 5th Intl. Conf. Autom. Face and Gesture Recogn.*, (Washington, DC), pp. 372–377, 2002.

[10] T. Feigl, C. Mutschler, and M. Philippsen, "Human Compensation Strategies for Orientation Drifts," in *25th IEEE Conf. Virtual Reality and 3D User Interfaces*, (Reutlingen, Germany), pp. 1–8, 2018.

[11] A. Mannini and A. Maria Sabatini, "Machine Learning Methods for Classifying Human Physical Activity from On-Body Accelerometers," *Sensors*, vol. 10, no. 1, pp. 1154–1175, 2010.

[12] T. Feigl, C. Mutschler, and M. Philippsen, "Head-to-Body-Pose Classification in No-Pose VR Tracking Systems," in *25th IEEE Conf. Virtual Reality and 3D User Interfaces*, (Reutlingen, Germany), pp. 1–2, 2018.

[13] D. Roth, C. Kleinbeck, T. Feigl, C. Mutschler, and M. E. Latoschik, "Beyond Replication: Augmenting Social Behaviors in Multi-User Social Virtual Realities," in *25th IEEE Conf. Virtual Reality and 3D User Interfaces*, (Reutlingen, Germany), pp. 1–8, 2018.

[14] J. Tiemann, F. Eckermann, and C. Wietfeld, "Atlas - an open-source tdoa-based ultra-wideband localization system," in *Intl. Conf. Indoor Positioning and Indoor Nav.*, (Alcala de Henares, Spain), pp. 1–6, 2016.

[15] E. Foxlin, "Head tracking relative to a moving vehicle or simulator platform using differential inertial sensors," in *Proc. SPIE Intl. Symp. AeroSense*, (Orlando, FL), pp. 133–144, 2000.

[16] G. Donato, V. Sequeira, and A. Sadka, "Stereoscopic helmet mounted system for real time 3D environment reconstruction and indoor ego-motion estimation," in *Proc. Symp. SPIE Defense and Security*, (Orlando, FL), pp. 1–12, 2008.

[17] E. Murphy-Chutorian, A. Doshi, and M. M. Trivedi, "Head pose estimation for driver assistance systems: A robust algorithm and experimental evaluation," in *Proc. IEEE Intelligent Transportation Systems Conf.*, (Bellevue, WA), pp. 709–714, 2007.

[18] T. Gadeke, J. Schmid, W. Stork, and K. Müller-Glaser, "Pedestrian dead reckoning for person localization in a wireless sensor network," in *Proc. Intl. Conf. Indoor Positioning and Indoor Nav.*, (Guimaraes, Portugal), pp. 1–4, 2011.

[19] L. Rong, Z. Jianzhong, L. Ming, and H. Xiangfeng, "A wearable acceleration sensor system for gait recognition," in *Proc. 2nd IEEE Conf. Industr. Electro. and Appl.*, (Harbin, China), pp. 2654–2659, 2007.

[20] A. R. Jiménez, F. Seco, J. C. Prieto, and J. Guevara, "Indoor pedestrian navigation using an INS/EKF framework for yaw drift reduction and a foot-mounted IMU," in *Proc. 7th Workshop Positioning Nav. and Comm.*, (Dresden, Germany), pp. 135–143, 2010.

[21] J. Windau and L. Itti, "Walking compass with head-mounted IMU sensor," in *Proc. 2016 IEEE Intl. Conf. Robotics and Automation*, (Stockholm, Sweden), pp. 5542–5547, 2016.

[22] S. Beauregard, "A helmet-mounted pedestrian dead reckoning system," in *Proc. 3rd Intl. Forum Applied Wearable Computing*, (Bremen, Germany), pp. 1–11, 2006.

[23] W. Kang, S. Nam, Y. Han, and S. Lee, "Improved heading estimation for smartphone-based indoor positioning systems," in *Proc. 23rd IEEE Intl. Symp. Personal, Indoor and Mobile Radio Comm.*, (Sydney, Australia), pp. 2449–2453, 2012.

[24] N. Roy, H. Wang, and R. Roy Choudhury, "I am a smartphone and I can tell my user's walking direction," in *Proc. 12th Intl. Conf. Mobile Systems, Appl., and Services*, (Bretton Woods, NH), pp. 329–342, 2014.

[25] J. Mantyjarvi, J. Himberg, and T. Seppanen, "Recognizing human motion with multiple acceleration sensors," in *Proc. IEEE Intl. Conf. Systems, Man, and Cybernetics*, (Tucson, AZ), pp. 747–752, 2001.

[26] D. Pai, I. Sasi, P. S. Mantripragada, M. Malpani, and N. Aggarwal, "Padati: A robust pedestrian dead reckoning system on smartphones," in *Proc. 11th Intl. Conf. Trust, Security and Privacy in Computing and Comm.*, (Liverpool, UK), pp. 2000–2007, 2012.

[27] N. Roy, *WalkCompass: Finding Walking Direction Leveraging Smartphone Inertial Sensors*. PhD thesis, Univ. of Shipbur, Columbia, SC, 2013.