AiRite: Towards Accurate & Infrastructure-Free 3-D Tracking of Smart Devices

Vivek Chandel
TCS Research & Innovation
Kolkata, India
vivek.chandel@tcs.com

Shivam Singhal
TCS Research & Innovation
Kolkata, India
s.singhal2@tcs.com

Avik Ghose
TCS Research & Innovation
Kolkata, India
avik.ghose@tcs.com

***Abstract*—Precise tracking of smart wearables in 3-D space is foundational in domains ranging from Augmented Reality (AR) and gesture recognition to robotics and indoor localization. Usually, a sensor-dependent approximation model, and a complementary infrastructure like acoustic, vision etc. is implemented alongside an onboard IMU (Inertial Measurement Unit) for an effective tracking. In this work, we present a novel 3-D tracking solution, 'AiRite' for commercial-grade smart wearables/mobiles using only their onboard IMU. Our tracking method mitigates the manifested inertial errors using a novel progressive zero correction, yielding superior results both for 2-D and 3-D trajectories. AiRite aims for a pervasive usage and is independent of device's form factor, or make of sensor enabling it to be used right out-of-the-box. The solution is shown to be more accurate than prior arts in tracking trajectories of basic shapes by moving the hand both in 2-D and 3-D. We depict trajectories of medium to long duration words written in air in a cursive handwriting using both smartwatch and smartphone. We further demonstrate robustness of AiRite by depicting panoramic 3-D texts written in air in a concave shape and visualizing the same from multiple angles, which are first results of their kind in this domain.**

***Index Terms*—MEMS, inertial sensors, tracking, wearable, smartphone**

I. Introduction

Consider a scenario where one is wearing a personal smartwatch, smart glove or holding a smartphone, and can digitally create 3-D text and shapes in one's own handwriting by just scribbling in air without any complementary hardware, or a case where hand gestures from a visually-challenged person can be tracked accurately & ubiquitously without the need of being in front of any camera or a special setup. Touchscreen inputs can be used for simple 2-D geometries, but lose their application in case of smartwatches, or if the trajectory is to be written/visualized in 3-D (e.g. creating CAD models of real-life 3-D objects by tracing object edges with a smart device [2]). True ubiquity in such cases can be achieved only if the solution is fully device-agnostic, and is independent of any hardware external to the device itself. In this work, we present such a 3-D tracking solution, 'AiRite', capable of running on smart wearables/devices with different form factors (wrist-watch, phone etc.) using solely the IMU chip present onboard these devices. *With a robust gravity removal, and highly effective yet frugal progressive zero-correction technique*, we are able to achieve a high degree of tracking accuracy in drawing shapes/gestures and in writing cursive words in air. To further strengthen the 3-D capability statement of our solution, we present tracking results of words written in a concave fashion so as to traverse all the 3 dimensions (Figure 1).

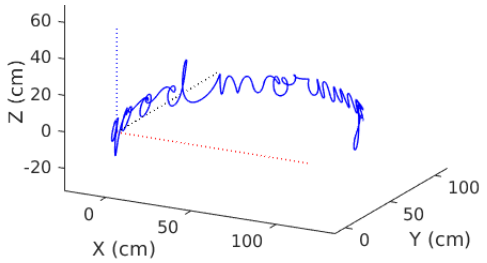

Fig. 1. Artistic 'good morning' text reproduced by AiRite in 3-D, written in air in a concave panoramic fashion while wearing a smartwatch.

The results presented are first of their kind to the best of our knowledge and prove that our system can provide extremely accurate short-duration 3-D trajectories using commercial-grade smart wearables in domains like healthcare, where potentially any part/joint of the body can be accurately tracked, e.g. unobtrusive tracking of range of motion using smartwatches in subjects with shoulder-injuries [3]. Our system can also immensely improve detection of fine hand activities [4] by providing detailed trajectories traversed by hand in air, and help creating highly accurate smartwatch-based biometric solutions by directly exploiting the 3-D trajectory data of wrist's motion [5][6].

II. Related Works

Errors manifested in IMU sensors have been attacked from various perspectives in prior researches. Pang et. al. [7] evaluated low-cost MEMS accelerometer at different rates of acceleration using a robotic arm and presented a study regarding different biases like random, thermal etc. Laotrakunchai et. al. [8] proposes an accelerometer based method, coupled with camera in order to determine the dimensions of objects in 3-D space.

There have been interesting attempts to use IMU-enabled wearables as devices to create texts/3-D gestures in air, which is also the major scope of this work. Arduser et. al. [9] uses smartwatch to classify 3-D accelerometer data to individual alphabets using DTW-based feature vector matching with

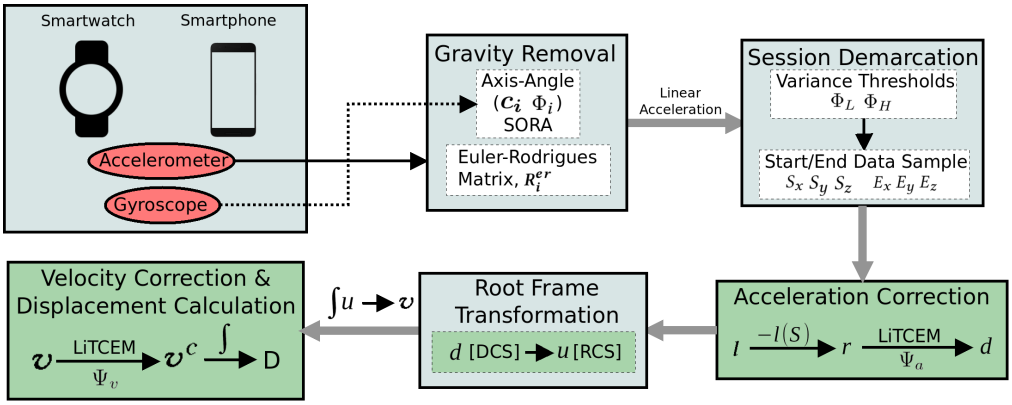

Fig. 2. Proposed Architecture

an accuracy of 94%. Amma et. al. [10] uses accelerometer and gyroscope attached to the back of the hand to detect motion while writing in air, and use an HMM to classify the motion data into individual characters. Agrawal et. al. [11] has presented some interesting results of air-writing where individual characters are written in air and are classified using grammar trees operating using individual strokes. Casanova et. al. [12] presents a bio-metric solution where the user performs an authentication by making a certain unique 3-D gesture in air using the device. Wang et. al. [13] proposes a digital pen employing accelerometer for recognizing handwritten digits and other 3-D gestures. All the mentioned works perform an error and/or gravity correction at some stage of their respective methods. Error correction and calibration for IMU sensors, specifically accelerometer, is also a widely researched area [14][15].

III. Foundational Challenges

In this section, we outline major challenges posed by a purely IMU-based tracking system.

A. Continuous Drifting

Various factors like cross-axis dependence, temperature, scaling factors etc. result in erroneous IMU data, which when numerically integrated (for velocity, displacement and angle of rotation) drifts the output by large factors. A common technique for drift correction is Zero Velocity Update using sensor-specific approximation error models, usually implemented on foot-mounted IMUs [16][17]. But the accuracy of the same is limited when the IMU is located on a device being moved by or attached to a hand.

B. Gravity Estimation

In order to accurately track the device's trajectory in 3-D space, an efficient filtering of gravity is required. Performing a traditional low-pass filtration on acceleration data, sometimes fused with orientation information from gyroscope can corrupt the numerical integrity of the data, and the resultant linear acceleration is also highly affected by the external motion being registered with the accelerometer.

C. Reference Coordinate System

Inertial sensors present onboard a device provide instantaneous data in a Device Coordinate System (DCS). While the device is in motion and changing its orientation continuously, the data needs to be converted from DCS to a single Universal Coordinate System (UCS), so that a meaningful vector analysis can be made on the same. Deriving the basis of such a coordinate system may require using a complementary sensor like a magnetometer, which presents challenges of its own, such as environmental disturbances calling for ambient calibrations. Our solution avoids the use of magnetometer to keep the overall system free from any additional calibration effort.

IV. Proposed Approach

We now present the details of our devised approach with a modular overview of our approach depicted in Figure 2. Our approach assumes a small interval where the hand is kept almost stationary immediately before and after the motion.

A. Gravity Removal

For estimating gravity component in the acceleration data, we propose a method where an initial gravity estimate is made from accelerometer, following which the subsequent estimates are generated solely using gyroscope. Let the gyroscope data at sample i be $\boldsymbol{r_i} = (r_i^x, r_i^y, r_i^z)$, and the accelerometer sample be $\boldsymbol{a_i} = (a_i^x, a_i^y, a_i^z)$. If the data is sampled at F_s samples a second, then the device's rotation from sample no. $i-1$ to i can be represented as $\boldsymbol{\theta_i} = ((\boldsymbol{r_{i-1}} + \boldsymbol{r_i})/2).dt$ where $dt = 1/F_s$.

Stacin et. al. [18] proposes that if the sampling frequency is high enough, then the rotation $\boldsymbol{\theta_i}$ can be approximated to an equivalent representation in axis and angle ($\boldsymbol{c_i}$ & Φ_i respectively) as follows:

$$\begin{aligned} \Phi_i = \|\boldsymbol{\theta_i}\|_2, \ and \ \boldsymbol{c_i} = \boldsymbol{\theta_i}/\Phi_i \\ \boldsymbol{c_i} = c_x\hat{i} + c_y\hat{j} + c_z\hat{k} \end{aligned} \tag{1}$$

An initial gravity estimate in DCS is made using the first stationary interval of duration d_g at the session's beginning

(d_g can be as small as $0.5s$). The initial gravity vector in DCS is then calculated as:

$$\boldsymbol{g_i} = \boldsymbol{g_1} \ \forall \ 1 \leq i \leq d_g.F_s \tag{2}$$

For $i > d_g.F_s$, the gravity vector $\boldsymbol{g_i}$ is transformed to $\boldsymbol{g_{i+1}}$ using Euler-Rodrigues rotation formula [19] as:

$$\boldsymbol{g_{i+1}} = \boldsymbol{R_i^{er}} \times \boldsymbol{g_i}, \ for \ i > d_g.F_s \tag{3}$$

where $\boldsymbol{R_i^{er}}$ is a special rotation matrix [19], and is derived using $\boldsymbol{c_i}$ and Φ_i.

Linear acceleration at i^{th} data sample, $\boldsymbol{l_i}$ can now be inferred simply as $\boldsymbol{l_i} = \boldsymbol{a_i} - \boldsymbol{g_i}$.

B. Session Demarcation

Demarcation stage is responsible for selecting an appropriate starting data sample, S from D_s and an ending data sample, E from D_e, where D_s and D_e represent inertial data of the stationary intervals immediately before and after the motion respectively. Since our system primarily targets motion by a human hand, keeping the device *perfectly* stationary before and after the motion is not practically feasible, hence a single best possible sample needs to be selected from a (relatively) noisy stationary interval. We propose following approach to select most appropriate boundary samples, which have been observed to yield excellent tracking results.

The measurement session's linear acceleration data $\boldsymbol{l}$=$(l_x\hat{i} + l_y\hat{j} + l_z\hat{k})$ as derived from previous stage, is divided into N data windows, with every window having K no. of data samples. Also let the standard deviations of three axes for i^{th} data window of acceleration be denoted as a vector $\boldsymbol{\Phi_i} = \Phi_i^x\hat{i} + \Phi_i^y\hat{j} + \Phi_i^z\hat{k}$ and let the j^{th} sample no. of i^{th} window be denoted as w_j^i. Any scalar-based logical operation on $\boldsymbol{\Phi_i}$ is applied on its individual components. Let the linear acceleration vector at $(w_j^i)^{th}$ sample be denoted as $\boldsymbol{l(w_j^i)}$. A low and high thresholds for the standard deviation of data windows are defined as Φ_L and Φ_H. It is observed that one fixed set of Φ_L & Φ_H performs satisfactorily with multiple devices.

Let n^{th} window be such that $\boldsymbol{\Phi_n} > \Phi_H$ and $\boldsymbol{\Phi_i} < \Phi_H \ \forall \ i < n$. Also let m^{th} window be such that $\boldsymbol{\Phi_m} < \Phi_L$ and $\boldsymbol{\Phi_i} > \Phi_L \ \forall \ m < i < n$. Then the starting sample no. for X-axis, S_x is calculated as:

$$S_x = argmin \ \{|l_x(w_i^n) - M_s|\} \ for \ (w_K^n/2) < i < w_K^n \tag{4}$$

$$where, \ M_s = \frac{\sum_{i=w_1^m}^{w_K^m} l_x(w_i^m)}{K}$$

Similar operation is carried out for S_y and S_z.

In order to end the session of motion, the accelerometer sensor is required to be in a stationary state for a certain duration. The proposed model works with duration as low as $0.5s$. Session's last data window (N^{th}) is such that $\boldsymbol{\Phi_N} < \Phi_L$. Now let p^{th} window be such that $\boldsymbol{\Phi_p} < \Phi_L$ & $\boldsymbol{\Phi_i} > \Phi_L \ \forall \ n < i < p$. Then the ending sample no. for X-axis, E_x is calculated as:

$$E_x = argmin \ \{|l_x(w_i^p) - M_e|\} \ for \ (w_K^p/2) < i < w_K^p \tag{5}$$

$$where, \ M_e = \frac{\sum_{i=w_1^N}^{w_K^N} l_x(w_i^N)}{K}$$

Similar operation is carried out for E_y and E_z.

In their essence, equations 4 and 5 pick up the sample point whose value is closest to mean of the data in their respective stationary intervals (D_s and D_e).

C. Modeling Error in Linear Acceleration

As mentioned in previous section, the difference in acceleration at samples S and E is the key to address the errors manifested in the gravity-corrected accelerometer data. We propose a method whereby the aforementioned parameter is modeled linearly in time domain, akin to performing a zero update. *This modeling is performed both on linear acceleration and derived velocity in a progressive fashion.*

In order to achieve this, first the linear acceleration, $\boldsymbol{l}$ is normalized with respect to the demarcated starting sample S. This eliminates any accumulated error in the accelerometer data when the data stream of the sensor has started. The normalized linear acceleration is calculated simply as:

$$r_x(i) = l_x(i) - l_x(S_x) \ for \ S_x \leq i \leq E_x \tag{6}$$

In a similar manner, $r_y(i)$ and $r_z(i)$ are also calculated to provide the complete $\boldsymbol{r(i)}$.

At this stage, the component $r_x(E_x)$ reflects the true value of the error in linear acceleration accumulated during the motion (along X-axis; similar operations and interpretations are made for Y and Z axes), mostly as a result of approximations in the angle-axis formulation. This error is then linearly distributed among the data before sample no. E_x as a time-varying parameter. We term this zero-correction model *Linear Temporal Cumulated Error Model (LiTCEM)*, which is frugal yet robust and accurate in performance, as is inferred from the evaluations discussed later.

The correction parameter used by the model is calculated as follows:

$$\Psi_x^a = (r_x(E_x))/(E_x - S_x) \tag{7}$$

Ψ_x^a and Ψ_x^a are calculated in a similar fashion using Y and Z axes components of $\boldsymbol{r}$ to give the final correction vector parameter as $\boldsymbol{\Psi^a} = \Psi_x^a\hat{i} + \Psi_y^a\hat{j} + \Psi_z^a\hat{k}$.

After $\boldsymbol{\Psi^a}$ has been calculated, its components are distributed among the components of $\boldsymbol{r}$ in a linear time-varying fashion, as also mentioned previously. The final corrected linear acceleration vector, $\boldsymbol{d}$ is then calculated as follows:

$$d_x(i) = r_x(i) - \delta_x^a(i) \tag{8}$$

where δ^a represents the temporal parameter calculated as:

$$\delta_x^a(i) = \Psi_x^a * (i - S_x) \ for \ (S_x + 1) \leq i \leq E_x$$

Similar operations are carried out for Y and Z axes to yield the corrected linear acceleration vector $\boldsymbol{d}$.

D. Re-orientation to Root Coordinate System

As previously discussed, the DCS acceleration $\boldsymbol{d}$ needs to be transformed to UCS, since the device continuously changes its orientation when the user freely moves their hand in air. Usually such a UCS is interpreted using magnetic north vector from magnetometer, and the gravity vector. But as already mentioned, magnetometer poses many calibration-based challenges when used in wild with varying magnetic fields impeding an accurate coordinate transformation, and hence affecting trajectory calculation. We hence propose a different UCS, a Root Coordinate System, RCS without using magnetometer at any stage, *whose basis is identical to the basis of the DCS at start of the stationary duration before the motion is performed* (samples 1 to $d_g.F_s$). Hence, RCS is equal to the DCS at the first sample of the duration d_g. Let $\lambda = d_g.Fs$.

In order to convert the acceleration vector $\boldsymbol{d(i)}$ which is in DCS, to the rotated vector $\boldsymbol{u(i)}$ in RCS, following is performed:

$$\boldsymbol{u(i)} = \boldsymbol{R^{er}_{i,1}} \times \boldsymbol{d(i)} \tag{9}$$

where $\boldsymbol{R^{er}_{i,1}}$ represents the Euler-Rodrigues rotation matrix for converting the vector from DCS at sample no. i to the DCS at starting sample, i.e., sample no. 1. This matrix is different from the matrix expressed in section IV-A, and can be derived from the same as follows:

$$\boldsymbol{R^{er}_{i,1}} = \boldsymbol{R^{er}_{\lambda}} \times \boldsymbol{R^{er}_{\lambda+1}} \times \boldsymbol{R^{er}_{\lambda+2}} \ldots \times \boldsymbol{R^{er}_{i-1}} \times \boldsymbol{R^{er}_{i}} \tag{10}$$

E. Velocity Correction

At this stage, the corrected acceleration $\boldsymbol{u}$ is available, and has already been rotated from DCS to RCS, which can be used for velocity calculation. The velocity vector, $\boldsymbol{v}$ is calculated by employing trapezoidal integration on $\boldsymbol{u}$ as following:

$$v_x(i) = \frac{u_x(i) + u_x(i-1)}{2 \times F_s} \; for \; (S_x + 1) \leq i \leq E_x \tag{11}$$

and similarly v_y and v_z to yield complete $\boldsymbol{v} = v_x\hat{i} + v_y\hat{j} + v_z\hat{k}$.

It is observed that $\boldsymbol{v}$ exhibits similar behavior in terms of error as $\boldsymbol{r}$. Hence, LiTCEM is performed on $\boldsymbol{v}$ as well to yield the corrected velocity $\boldsymbol{v^c}$ as follows:

$$v^c_x(i) = v_x(i) - \delta^v_x(i) \tag{12}$$

where δ^v represents the temporal parameter for velocity, and is calculated as:

$$\delta^v_x(i) = \Psi^v_x * (i - S_x) \; for \; (S_x + 1) \leq i \leq E_x$$

and Ψ^v is similar to Ψ^a and is calculated as:

$$\Psi^v_x = (v_x(E_x))/(E_x - S_x) \tag{13}$$

v^c_y and v^c_z are calculated in a similar fashion giving the final corrected velocity vector as $\boldsymbol{v^c} = v^c_x\hat{i} + v^c_y\hat{j} + v^c_z\hat{k}$.

Once we have the corrected velocity, estimating displacement vectors is trivial. Trapezoidal integration of $\boldsymbol{v^c}$ yields the final displacement, $\boldsymbol{D}$ as follows:

$$D_x(i) = \frac{v^c_x(i) + v^c_x(i-1)}{2 \times F_s} \; for \; (S_x + 1) \leq i \leq E_x \tag{14}$$

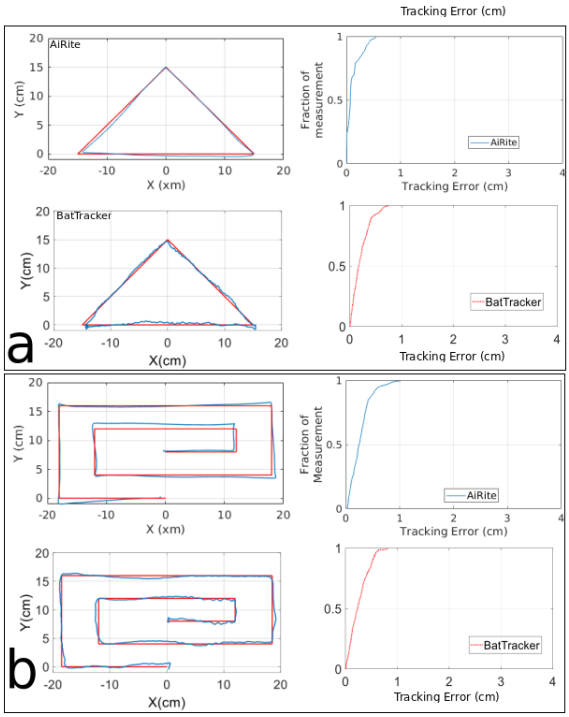

Fig. 3. For each of a and b, first row corresponds to trajectory produced by AiRite (red: ground truth, blue: calculated trajectory) on left, and the corresponding CDF of error distribution on the right. Second row corresponds to the corresponding output along with its CDF as produced by BatTracker [20] (figures directly imported from the paper [20] for comparison purpose due to lack of vector data).

V. AiRite in Action

In this section we evaluate our error model's performance using AiRite for 2-D and 3-D scenarios and present some interesting results. For smartwatch, we use an LG Urbane running Wear OS 2. We also present results for AiRite running on a smartphone form factor to prove seamless performance of our system across devices. A data sampling rate of 80Hz is used for both accelerometer and gyroscope sensors. As also mentioned previously, the user holds the hand stationary (as per convenience) for a short duration ($\approx 0.5s$) immediately before and after the writing action is performed. For every writing activity, user assumes the corner of smartphone as the writing tip, while wearing a smartwatch simultaneously on their wrist. The writing motion is performed leveraged at the elbow (instead of wrist) so that correct acceleration signals are registered with the smartwatch. All trajectories in the following sections are smartwatch-derived (except Figure 4(b)). The duration of writing the words varied from $\approx 5s$ for short words, up to $13s$.

A. 2-D Trajectories

Here, we examine AiRite's performance in drawing trajectories in 2-D ranging from simple shapes to complex long words written in cursive handwriting.

1) Basic Shapes: BatTracker, as devised by Zhou et. al. [20] presents a 3-D tracking method using a combination of

Fig. 4. Words digitally reproduced by AiRite, along with ground truth above each reproduction. (a) LG Urbane Smartwatch, and (b) Oneplus 5 Smartphone. Starting and ending points of tracing for a word are marked by green and red dot respectively.

IMU and acoustic techniques, where the authors have shown it to be more accurate than a similar CAT system devised by Mao et. al. [21], and AAMouse introduced by Yun et. al. [22]. We compare AiRite system directly with BatTracker.

Figure 3 shows the trajectories of triangle and loop drawn using AiRite and BatTracker, with the respective error distributions. AiRite presents better accuracy for triangle. For loop, the 90th percentile error equals BatTracker ($0.5cm$). It is noteworthy that unlike BatTracker, AiRite uses only device's onboard IMU, and doesn't require any extensive pre-calibration of the system.

2) Words in Cursive Handwriting: In order to perform this experiment, a word was written on a large sheet of paper, and the hand was moved along the trajectory of the word. Figure 4 shows the original word to be traced, and screenshots of the respective trajectory produced by AiRite on smartwatch (Figure 4(a); trajectory plotted on a computer) and smartphone (Figure 4(b)). RCS was initialized such that the trajectories are drawn in the plane perpendicular to gravity (RCS' XY plane). The maximum displacement error accumulated in the Z-axis of RCS was only $12mm$. Mean error in the XY plane bounding box dimensions of the calculated trajectories was found to be $3.4cm$.

B. 3-D Trajectories

Finally we analyze AiRite's performance by moving the device unconstrained in 3-D space. We again compare AiRite with BatTracker system while drawing the double circle geometry with some elevation with the XY plane, so that the device moves in all the 3 axes. Figure 5 shows the trajectory reproduced by AiRite and BatTracker for similar geometry. It can be inferred from the error distribution functions (Figure 5(b)) that AiRite outperforms BatTracker in tracking the

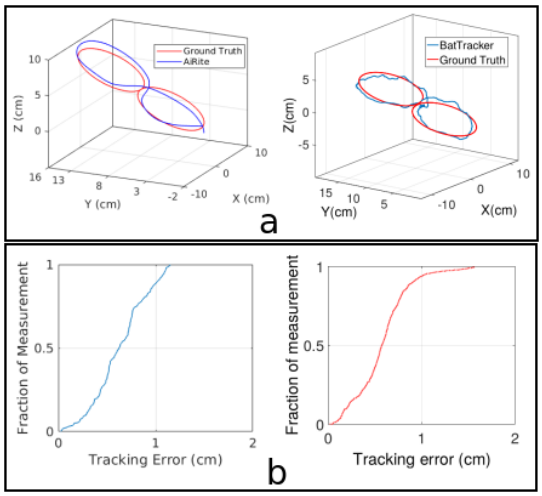

Fig. 5. (a): 3-D trajectories produced by AiRite (left) and BatTracker (right). (b): CDF of error distribution in drawing the 3-D geometry for AiRite (left) and BatTracker (right). CDF plots are shown separately due to lack of vector data for BatTracker. Figures directly imported from the paper [20] for comparison purpose.

geometry, with a maximum trajectory error of $1.1cm$ against BatTracker's $\approx 1.5cm$.

*1) **Curved Panoramic Texts**:* The method underlying AiRite system is capable of projecting trajectories from DCS to a unique RCS in 3-D, when the device is moved in random orientations, which opens up possibilities of generating more creative and complex geometries in air, than just simple geometries. As a start, we tested the application's performance by writing panoramic texts, where the user wrote a text in air wearing a smartwatch while sitting on a chair starting

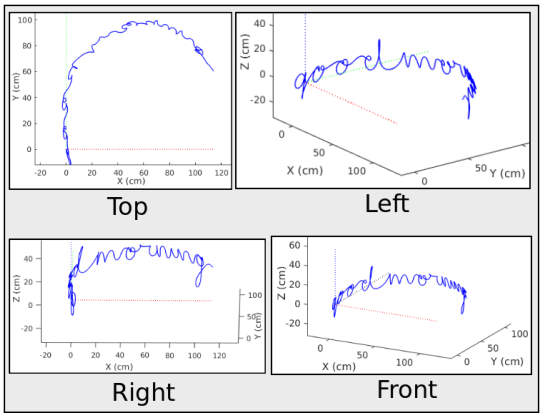

Fig. 6. Curved panoramic text 'goodmorning' written in air using a smartwatch viewed from different angles.

from left of their head and continuing around the head in a circular concave fashion. An example result is illustrated in Figure 6 for the word **'goodmorning'**. The trajectories in Figure 6 prove that DCS to RCS transformation of our system works effectively, in series to the LiTCEM progressive zero correction.

VI. Conclusion and Future Directions

In this work we presented a solution capable of accurately tracking smart wearables (and potentially any commercial device with an IMU) in 3-D. Proposed solution was shown to perform effectively in tracking the hand in writing short to medium duration cursive words both in 2-D and 3-D, without any pre-calibration or complementary hardware, and on devices with form factors of both wrist-watch and phone.

Currently, the solution is being developed as a highly accurate system for recognizing complex hand gestures, and also for a low-cost and ubiquitous gait analysis of Parkinsons' patients using commercial off-the-shelf IMUs on wrists and feet, with detailed and precise calculation of stride trajectories and hand tremors.

References

[1] V. Chandel and A. Ghose, "Airite: Infrastructure-free cursive writing and drawing in air using smart devices," in *Adjunct Proceedings of the 2019 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2019 ACM International Symposium on Wearable Computers*, ser. UbiComp/ISWC '19 Adjunct. New York, NY, USA: ACM, 2019, pp. 257–259. [Online]. Available: http://doi.acm.org/10.1145/3341162.3343839

[2] C. P. Jones, J. Lenchner, N. Masters, D. A. Mazzella, J. A. Oravec, and R. A. Rey, "Creating three dimensional models with acceleration data," May 29 2018, uS Patent 9,984,502.

[3] J. N. Côté, D. Raymond, P. A. Mathieu, A. G. Feldman, and M. F. Levin, "Differences in multi-joint kinematic patterns of repetitive hammering in healthy, fatigued and shoulder-injured individuals," *Clinical Biomechanics*, vol. 20, no. 6, pp. 581–590, 2005.

[4] G. Laput and C. Harrison, "Sensing fine-grained hand activity with smartwatches," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. ACM, 2019, p. 338.

[5] I. Griswold-Steiner, R. Matovu, and A. Serwadda, "Handwriting watcher: A mechanism for smartwatch-driven handwriting authentication," in *2017 IEEE International Joint Conference on Biometrics (IJCB)*. IEEE, 2017, pp. 216–224.

[6] J. Yang, Y. Li, and M. Xie, "Motionauth: Motion-based authentication for wrist worn smart devices," in *2015 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*. IEEE, 2015, pp. 550–555.

[7] G. Pang and H. Liu, "Evaluation of a low-cost mems accelerometer for distance measurement," *Journal of Intelligent and Robotic Systems*, vol. 30, no. 3, pp. 249–265, 2001.

[8] S. Laotrakunchai, A. Wongkaew, and K. Patanukhom, "Measurement of size and distance of objects using mobile devices," in *Signal-Image Technology & Internet-Based Systems (SITIS), 2013 International Conference on*. IEEE, 2013, pp. 156–161.

[9] L. Ardüser, P. Bissig, P. Brandes, and R. Wattenhofer, "Recognizing text using motion data from a smartwatch," in *2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*. IEEE, 2016, pp. 1–6.

[10] C. Amma, M. Georgi, and T. Schultz, "Airwriting: Hands-free mobile text input by spotting and continuous recognition of 3d-space handwriting with inertial sensors," in *Wearable Computers (ISWC), 2012 16th International Symposium on*. IEEE, 2012, pp. 52–59.

[11] S. Agrawal, I. Constandache, S. Gaonkar, R. Roy Choudhury, K. Caves, and F. DeRuyter, "Using mobile phones to write in air," in *Proceedings of the 9th international conference on Mobile systems, applications, and services*. ACM, 2011, pp. 15–28.

[12] J. G. Casanova, C. S. Ávila, A. de Santos Sierra, G. B. del Pozo, and V. J. Vera, "A real-time in-air signature biometric technique using a mobile device embedding an accelerometer," in *International Conference on Networked Digital Technologies*. Springer, 2010, pp. 497–503.

[13] J.-S. Wang and F.-C. Chuang, "An accelerometer-based digital pen with a trajectory recognition algorithm for handwritten digit and gesture recognition," *IEEE Transactions on Industrial Electronics*, vol. 59, no. 7, pp. 2998–3007, 2012.

[14] J. Pan, C. Zhang, and Q. Cai, "An accurate calibration method for accelerometer nonlinear scale factor on a low-cost three-axis turntable," *Measurement Science and Technology*, vol. 25, no. 2, p. 025102, 2014.

[15] P. Batista, C. Silvestre, P. Oliveira, and B. Cardeira, "Accelerometer calibration and dynamic bias and gravity estimation: Analysis, design, and experimental evaluation," *IEEE transactions on control systems technology*, vol. 19, no. 5, pp. 1128–1137, 2011.

[16] R. Zhang, H. Yang, F. Höflinger, and L. M. Reindl, "Adaptive zero velocity update based on velocity classification for pedestrian tracking," *ieee Sensors journal*, vol. 17, no. 7, pp. 2137–2145, 2017.

[17] S. K. Park and Y. S. Suh, "A zero velocity detection algorithm using inertial sensors for pedestrian navigation systems," *Sensors*, vol. 10, no. 10, pp. 9163–9178, 2010.

[18] S. Stančin and S. Tomažič, "Angle estimation of simultaneous orthogonal rotations from 3d gyroscope measurements," *Sensors*, vol. 11, no. 9, pp. 8536–8549, 2011.

[19] J. S. Dai, "Euler–rodrigues formula variations, quaternion conjugation and intrinsic connections," *Mechanism and Machine Theory*, vol. 92, pp. 144–152, 2015.

[20] B. Zhou, M. Elbadry, R. Gao, and F. Ye, "Battracker: high precision infrastructure-free mobile device tracking in indoor environments," in *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*. ACM, 2017, p. 13.

[21] W. Mao, J. He, and L. Qiu, "Cat: high-precision acoustic motion tracking," in *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*. ACM, 2016, pp. 69–81.

[22] S. Yun, Y.-C. Chen, and L. Qiu, "Turning a mobile device into a mouse in the air," in *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 2015, pp. 15–29.