Indoor Localisation and Navigation on Augmented Reality Devices

Gaurav Gupta* Nishant Kejriwal Prasun Pallav Ehtesham Hassan Swagat Kumar
Ramya Hebbalaguppe
TCS Innovation Labs, New Delhi, India

ABSTRACT

We present a novel indoor mapping and localisation approach for Augmented Reality (AR) devices that exploits the fusion of inertial sensors with visual odometry. We have demonstrated the approach using Google Glass (GG) and Google Cardboard (GC) supported with an Android phone. Our work presents an application of Extended Kalman Filter (EKF) for sensor fusion for AR based application where previous work on Bag of Visual Words Pairs (BoVWP) [10] based image matching is used for bundle adjustment on Fused odometry. We present the empirical validation of this approach on three different indoor spaces in an office environment. We concluded that vision complimented with inertial data effectively compensate the ego-motion of the user, improving the accuracy of map generation and localisation.

Index Terms: Human-centered computing [Interaction paradigms]: Mixed / augmented reality—;Mathematics of computing [Probabilistic reasoning algorithms]: Kalman filters—

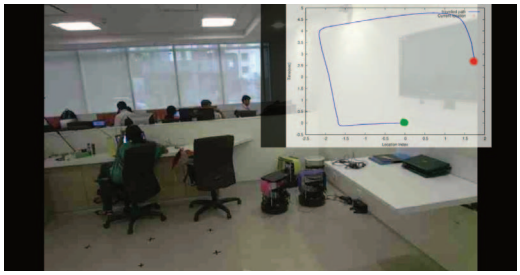

Figure 1: Dynamically generated map during the exploration of a new indoor environment. The map is displayed on AR device's screen (See: top-right). Observe the green dot that corresponds to the starting position of person and the red dot corresponding to his current location on the dynamic map.

1 INTRODUCTION

A dynamic indoor localisation system is necessary to assist a person navigating inside a building particularly when the person is exploring the building first time. In such a scenario, a wearable, location provider tool with good precision proves to be very important aid. Such tools have application in many places such as for assistive tours in an industry setting, museums and art galleries. Indoor localisation and navigation in these GPS-denied environment has been a challenging problem for robotics and vision researchers. There has been several Simultaneous Localization and Mapping (SLAM) methods for indoor environment which include: use of indoor local references i.e. visual landmark points [7], WiFi/ bluetooth based localisation and Inertial Measurement Unit(IMU) [22]. Individually different sensors exploit on unique characteristics for task. WiFi/bluetooth based approach use the intensity of received signal and fingerprinting for locating where more calibrated access points return improved accuracy. IMU based positioning utilize sensor fusion on accelerometer, gyroscope and compass recording for localisation [6]. The gyroscope provides a very good dynamic response measuring the change in the device orientation whereas compass gives the device orientation with respect to the North direction.

However, the accelerometer values based on IMU of a wearable/hand-held AR device are very sensitive to the body and device pose, and movement because of walking. A massive error gets accumulated in the distance values after few steps from start point. This requires periodic measurements from sensors to correct the accumulated drift in position values. In addition to IMU data, the robotics researcher have also applied vision and laser based motion detection techniques for SLAM. However applicability of vision sensor for SLAM is subject to the computational power and battery life available in the device. In particular, the focus of this work: AR devices such as GG and its frugal alternatives like GC and Wearality supported with common smartphones does not have enough hardware support for heavy processing. In this direction, our work presents a novel sensor fusion framework for SLAM combining the observations from vision with observations from IMU (An instance illustrated in Figure 1), which uses a client-server architecture for addressing the computationally intensive vision processing. The key contributions of our work are summarized below.

1 Sensor fusion model for AR devices for SLAM application using Visual odometry and IMU generated acceleration data: We use EKF based formulation for correction in egocentric motion and accurate pose estimation for map generation and localisation.

2 Server-client framework for sensor fusion based map generation: we present an interface between the backend server and wearable AR device for obviating the dependency on limited processing capability [17] of the device. This method is a form of pervasive computing and hence can be useful for processing any real time image processing based algorithm on a remote hardware, employing an AR device only for input, output and processing computationally inexpensive tasks. We demonstrate this framework for real-time map generation and localisation.

Rest of the paper is organised as follows: In Section 2, we present the literature survey of SLAM for indoor environment. Section 3 is divided into four sub parts: In Section 3.1, we address mechanism of data transport between the AR device and the back-end; section 3.2 describes the application at the client end where AR wearable is used; then section 3.3 gives an overview of localisation method at server end. Section 4 describes the localisation method in detail, including visual odometry, inertial odometry, sensor fusion and bundle adjustments. Section 5 details the results with experimental set-up and discussion. Lastly, section 6 deals with conclusions and avenues for future work.

*e-mail: g.gupta7@tcs.com

978-1-5090-3740-7/16 $31.00 © 2016 IEEE
DOI 10.1109/ISMAR-Adjunct.2016.45

IEEE computer society

2 Related Work

There have been previous attempts made for indoor localisation using smartphones with built in cameras [11], [20] and [18]. Here, smartphone serves the purpose of indoor localisation just like the outdoor navigation using GPS. However, using the image-based matching approach, the user has to always point the phone camera towards field of view(FoV) [20]. Other image-based localisation methods were tried on human operated backpack system [12] [13]. Though, a hands-free AR device will truly serve the purpose of a useful and dynamic indoor navigation. In past couple of years, the development on AR has triggered many applications, for example - in activity recognition using eye blink detection and head movement tracking [9], face recognition for improvement in social interaction [17] and gesture recognition [16] for hands-free motion interaction. The only bottleneck is the limited processing capability of many frugal wearable AR devices. This issue has been reported and addressed efficiently in our paper.

For localising a person in a predefined environment, learning landmarks with their respective positions can give an accurate location while testing [7]. In more practical scenarios where user explores an unknown indoor environment, we need an approach that works without any prior information about indoor setting. Use of IMU , communication modules, camera, SONAR or LIDAR are sensors which can be used here. Woodman and Harle [22] utilize a foot-mounted inertial unit, a detailed building model, and a particle filter combined to provide absolute positioning, despite the presence of drift in the inertial unit and without knowledge of the user's initial location. This approach is not useful when the data generation unit is a part of user's head-mount where jerks of his steps can not be reflected. Also the inertial units, particularly accelerometer is very noisy to use alone for position estimation [23]. Another approach includes camera based SLAM, which has been studied widely in recent research [20], [12], [19] , [15]. RGB camera image acquired through AR device proves to be an inexpensive and more effective approach compared to the other sensors aforementioned; making it more informative and suitable for place recognition. Its downside is the drift and occasional high noise. In our work the camera inputs are used together with inertial data to reduce both the drift and noise.

Corke et al [3] have discussed fundamentals of inertial and visual sensor data in motion estimation and 3D reconstruction applications. Fusion needs a non-linear motion model for tracking user's location [8] [1]. Hol et al [8] use methods of Extended Kalman Filter, Unscented Kalman Filter and particle filter for motion model. The rotation and translation sensor calibration between the two sensors is discussed in [14].

These fusion approaches are used mainly for robotics applications for localising a bot which exhibits a simple motion, unlike our approach which addresses the egocentric constraints of head motion and arbitrary walking patterns. We have developed a real time system that shows map overlay on device's screen and updates it at every 200*ms*. The complete approach is defined in rest of the paper.

3 Proposed Method

We have proposed a localisation and navigation application using an AR device such as GG and GC in conjunction with an Android smartphone. Figure 2 illustrates the client-server framework which has three vertical components. The client is basically an android wearable/hand-held device which is used only to fetch the scene-images in real-time, and the IMU data. It also serves as an interactive console for the user by showing the dynamic location of the navigator in the generated map. The server which is remotely located contains the processing unit for performing all computations on image and sensor data. The central vertical maintains the two-way communication between client and server.

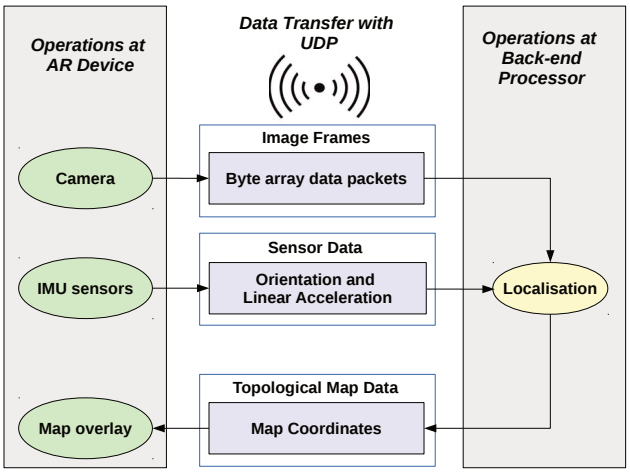

Figure 2: Detailed data flow graph on User Datagram protocol (UDP). The server and client communication happens over the WiFi in our proposed framework. The ellipses depict the processing units and the rectangles are data transmitted over network.

3.1 Data Transport between Client and Server

Figure 2 shows the two way communication between server and client uses User Datagram Protocol (UDP) based data transmission over WiFi. The transmission should be computationally simple and fast enough to avoid any added unnecessary lag in reporting new location index. Against other common transport layer protocol UDP is simple protocol with least processing which doesn't involve any handshake mechanism. Though it is unreliable and does not check packet drops, its fast mechanism with minimal network latency and bandwidth overhead works best for us.

The host address and UDP port number of server is used to link the client. Once the connection is established, client starts transmitting live stream of Data packets. Each packet is a maximum 64KB in size which contains - Image frame index, Image data, Acceleration vector and Orientation vector. On the other side, server takes the packet input at this port and transmit Image frame index and map coordinates in single packet at the same port. UDP exchange about $80-100$ packets/sec. in the tested experiment set-up (see section 5.1). The AR device requires quick response from server, hence ephemeral packet drops is a second concern over transmission speed. The *image frame index* keeps the sent and received packet in synchronization in situations of drops in packets.

3.2 Operations at AR Device - Client side

The operations at wearable device comprise of three main modules as shown in Fig. 2-

- *OpenCV camera* fetches frames from camera and passes it over to UDP transmission thread. It passes JPEG encoded image frame along with the image metadata (Image frame index) to the data packet.
- *IMU sensor* generates acceleration (in the 2D plane of user's path) and orientation sensor data to transmit over the network.
- *Map overlay* receives the indoor map coordinates from the server for displaying on GG screen. The point coordinates are overlayed as a topological map using line plot functionality in OpenCV[1] library.

[1]http://opencv.org/

3.3 Operations at Backend Server

The operations at back-end server starts with subscribing data from the defined UDP host address and port. The localisation module running of the server is responsible for all the computation steps which includes odometry generation, sensor fusion, and bundle adjustment for image based loop closure detection.

Figure 3 illustrates the flow-chart of the processing in localisation module. The Visual Odometry makes use of Shi Tomasi descriptors [21], tracked over time by Lucas Kanade's optical flow [2]. I_k is the input frame at time k. The sensor data (acceleration) is used to complement visual odometry using EKF based sensor fusion which gives a combined decision on pose estimation. *Condition (A)* holds true only when current position is greater then previous position by atleast Δ; where Δ is computed empirically – is the least distance between previous and current landmark where loop closure is not detected. Finally, loop closure detection is demonstrated as shown in the last block. The detailed explanation for these is given in the Section 4.

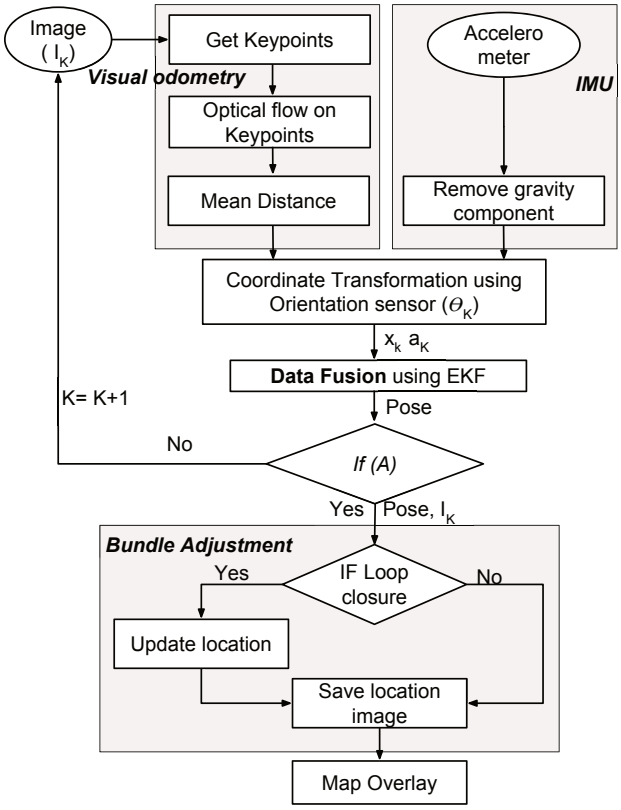

Figure 3: Flow-chart of computation steps in localisation module running on server side.

4 Computations Performed on Server Side

4.1 Visual Sensor Odometry

In this step, we generate the visual odometry from the camera ego-motion. We measure the distance traveled by the user from the initial point.

Optical flow represents the motion of key-points (like objects, edges or corners) in an image. If the camera is moving forward, the key-points in the image plane will move outwards from focus of expansion in the image plane. Similarly, if camera is going backward, the key-points will move inwards to the same focus in the image plane. Considering this behavior, we have used the focus of expansion at image plane center and calculated the distance of each descriptor from the center. Number of descriptors depends on the frequency content in the scene. Hence, the mean distance $dist_z^c$ of the key-points from the focus is calculated per frame to make scene invariant odometry. The magnitude of keypoints' motion vectors is directly proportional to camera's velocity and hence the distance traveled by camera is represented by $dist_z^c$. Lukas kanade [2] method is used for optical flow implementation and Shi Tomasi descriptor [21] for computing keypoints. The parameters of optical flow are tuned for optimal performance.

$$dist_z^c \propto \frac{\Sigma_{i=1}^{N} \parallel P_i - C \parallel_2}{N} \tag{1}$$

In Equation 1, P_i is the i^{th} keypoint, N is the total number of keypoints, C is the focus center, and $dist_z^c$ is distance traveled by user along z axis of camera coordinate system(**c**) (refer section 4.3) from initial point. The equation is computed using empirically derived factor of $\frac{1}{10}$.

We generate new key-points in image based on their drop in population since previously initialised. Refer Figure 3 for Key-point initialization step of visual odometry. This step is performed when the first image or if number of key-points drops by atleast $1/4^{th}$ of its original number. The fraction is derived empirically for optimal performance. Subsequently, the optical flow helps in finding the distance traveled by the camera from the reference point.

4.2 Inertial Sensor Odometry

AR wearable in use should include Accelerometer, Gyroscope and Compass for correctly defining IMU sensor data. Android operating environment implements method of IMU sensor fusion [6] and uses these three sensors to derive device's accurate orientation and acceleration. The acceleration derived from fusion is without the gravity acceleration component. Also the derived orientation has less noise and reduced gyro drift. TYPE_LINEAR_ACCELERATION module from Android Sensor API [4] is used directly as the measurement input from IMU. Orientation value from TYPE_ORIENTATION module of Android Sensor API [5] is used for coordinate transformation. Refer equation 2 and 3.

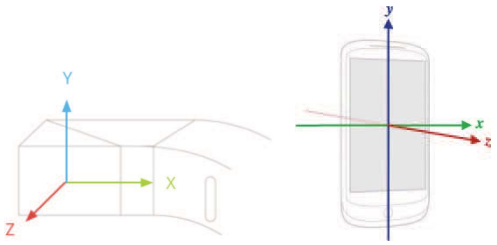

Figure 4: (a) depicting the axes of GG coordinate system and (b) corresponds to the phone coordinate system. Both are referenced from device's screen.

4.3 Coordinate System Transformation

Figure 4 shows that reference of the camera coordinate system(**c**) is set with respect to device's screen in default orientation. Camera and inertial sensor have the same coordinate system as the device. Accelerometer's (acc_z^c) and image plane's ($dist_z^c$) reading in $-Z$ axis of device is used as measurement from IMU and visual odometry respectively. θ^{wc} is the horizontal plane (**P**) orientation of Camera coordinate system (**c**) with respect to earth coordinate system (**w**). The map has been plotted on 2D plane (**P**) as shown in **w** of Figure 5

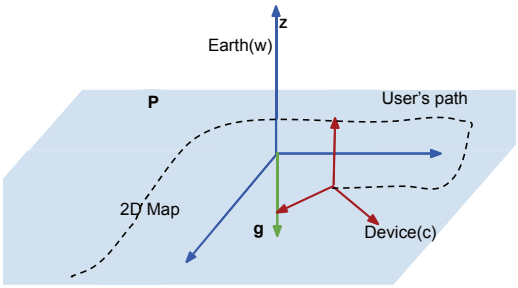

Figure 5: The transformation from the device coordinates system (blue) to the earth coordinates system (red) is shown. Equations 2, 3 show the transformations.

. After simplification, the transformation from **c** to **w** is represented in Equations 2 and 3.

$$\begin{bmatrix} pos_x^w \\ pos_y^w \end{bmatrix} = \begin{bmatrix} -cos(\theta^{wc}) \\ -sin(\theta^{wc}) \end{bmatrix} dist_z^c \tag{2}$$

$$\begin{bmatrix} acc_x^w \\ acc_y^w \end{bmatrix} = \begin{bmatrix} -cos(\theta^{wc}) \\ -sin(\theta^{wc}) \end{bmatrix} acc_z^c \tag{3}$$

4.4 Motion Model and Sensor Fusion

Acceleration received from IMU sensor is used to complement the visual odometry for more accurate pose estimation. To fuse Visual odometry and sensor odometry data, an EKF framework is used which solves for Non linear motion model of user.

The state transition and measurement equation is defined as-

$$\mathbf{S}_k = f(\mathbf{S}_{k-1}) + w_k \tag{4}$$

$$\mathbf{Z}_k = h(\mathbf{S}_k) + v_k \tag{5}$$

where w_k is process noise and v_k is measurement noise.

$\mathbf{S}_k \in \mathbb{R}^{2x1}$ is the State vector which is representing the motion variables of user. It is defined as-

$$\mathbf{S}_k = [\mathbf{x}_k \ \ \mathbf{v}_k \ \ \mathbf{a}_k]^T \tag{6}$$

$\mathbf{x}_k$ Position $[m]$
$\mathbf{v}_k$ Velocity $[m/s]$
$\mathbf{a}_k$ Acceleration $[m/s^2]$

The measurement matrix ($\mathbf{Z}_k$) represents the observed data from Visual odometry and inertial sensor of device. Where position from Visual odometry ($\mathbf{x}_k^{vis}$) and acceleration from inertial sensor ($\mathbf{a}_k^{imu}$) is used directly as measurement input.

$$\mathbf{Z}_k = [\mathbf{x}_k^{vis} \ \ \mathbf{a}_k^{imu}]^T \tag{7}$$

where $\mathbf{x}_k^{vis} = (pos_x^w, pos_y^w)$ and $\mathbf{a}_k^{imu} = (acc_x^w, acc_y^w)$ from equation 2 and 3.

The state transition is non linear equation. It is approximated by first order Taylor series expansion around the nearest linearisation point.

$$f(\mathbf{S}_{k-1}) = f(\mathbf{S}_{k-1}^o) + \frac{\partial(f(\mathbf{S}_{k-1}^o)}{\partial(\mathbf{S}_{k-1}^o)}(\mathbf{S}_{k-1} - \mathbf{S}_{k-1}^o) \tag{8}$$

The nearest linearisation point $\mathbf{S}_{k-1}^o$ is $\mathbf{S}_{k-2}$ and it implies-

$$\mathbf{S}_{k-1} = f(\mathbf{S}_{k-1}^o) \tag{9}$$

The partial derivative is the Jacobian which is written as-

$$F = \begin{bmatrix} \mathbf{I} & \Delta T\mathbf{I} & \Delta T^2\mathbf{I} \\ 0 & \mathbf{I} & \Delta T\mathbf{I} \\ 0 & 0 & \mathbf{I} \end{bmatrix} \tag{10}$$

F is function of time interval between successive measurements. The state transition equation is presented is Prediction equation. On the other hand, we have assumed measurement equation is linear and H is measurement.

$$H = \begin{bmatrix} \mathbf{I} & 0 & 0 \\ 0 & 0 & \mathbf{I} \end{bmatrix} \tag{11}$$

The Prediction is done as:

$$\mathbf{S}_k^- = \mathbf{S}_{k-1} + F(\mathbf{S}_{k-1}) - F(\mathbf{S}_{k-2}) \tag{12}$$

$$P_k^- = FP_{k-1}F^T + Q \tag{13}$$

The error co-variance matrix R corresponds to v_k represents the confidence of respective odometries. We assume their noise components are uncorrelated to each other. Hence it will be a diagonal matrix.

$$R = \begin{bmatrix} \sigma^{vis} & 0 \\ 0 & \sigma^{inr} \end{bmatrix} \tag{14}$$

The innovation is done as-

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1} \tag{15}$$

$$\mathbf{S}_k = \mathbf{S}_k + K_k(\mathbf{Z}_k - H\mathbf{S}_k^-) \tag{16}$$

$$P_k = (\mathbf{I} - K_k H)P_k^- \tag{17}$$

After several several examination using trial and error approach of mapping, the error covariances σ^{vis} and σ^{inr} are used as 0.0001 and 0.001.

4.5 Bundle Adjustments using Loop Closure Detection

Bundle Adjustments aims at minimizing the accumulated error in user map trajectory over time. In this paper, we take help of loop-closure detection for bundle adjustment. At loop closure, the final fused odometry gets updated with the odometry of loop closured past scene. The previous odometry of the same place has less accumulated error involved. Hence landmark images from user's FoV

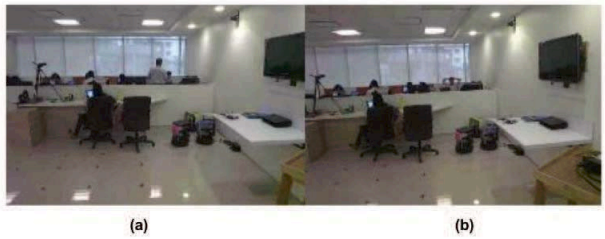

Figure 6: Scene (a) which depicts the previously visited landmark is matched with Scene (b) the current scene. Notice that a loop closure is detected even with the dynamic changes such as in the scenario where people in the scene move.

are saved and used as reference images. We are using method of loop closure detection from [10], where a robust technique of image representation is defined using Bag of Visual Words Pair(BoVWP) with SURF descriptors. A matching coefficient is set for controlling image matching between query image and reference image. If this coefficient is greater than a certain threshold, the query image is confirmed to be a loop closure otherwise a new landmark is created in the map overlay.

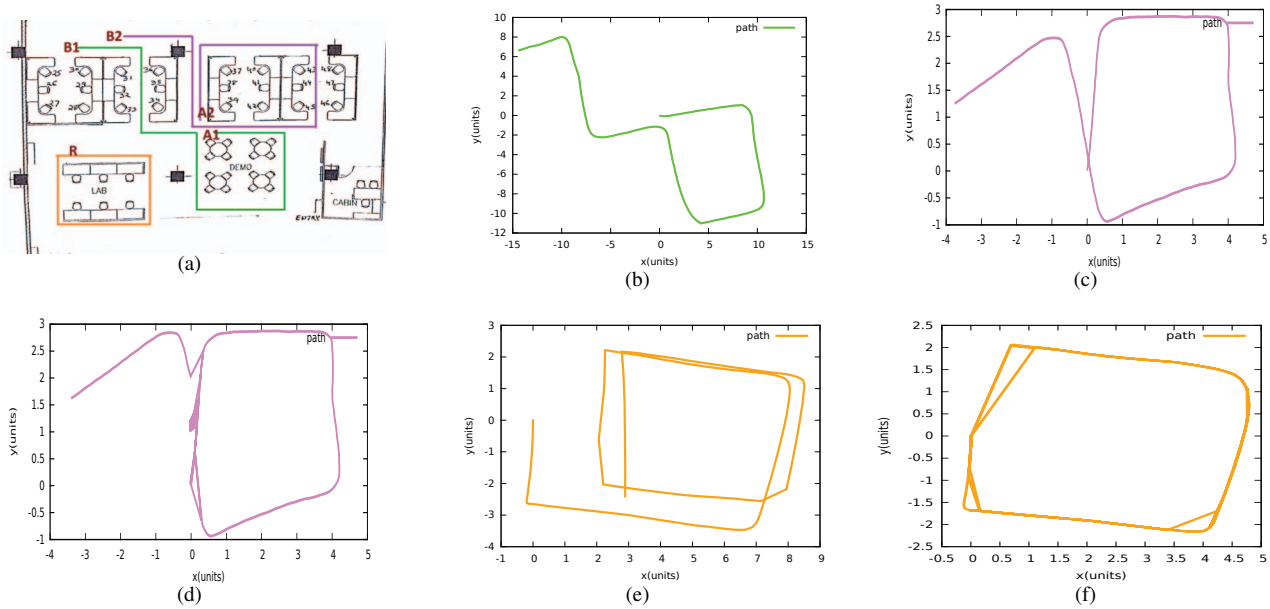

Figure 7: Map overlay on GG display. (a) Experimental setup - top view of the indoor office hall where A1-B1, A2-B2 and R is the ideal path for user. (b) Output map of traversal on A1-B1. It doesn't include any landmark revisit. (c) Output map of traversal on A2-B2 without bundle adjustment. (d) Output map of traversal on A2-B2 with bundle adjustment. (e) Output map of traversal on R without bundle adjustment. (f) Output map of traversal on R with bundle adjustment.

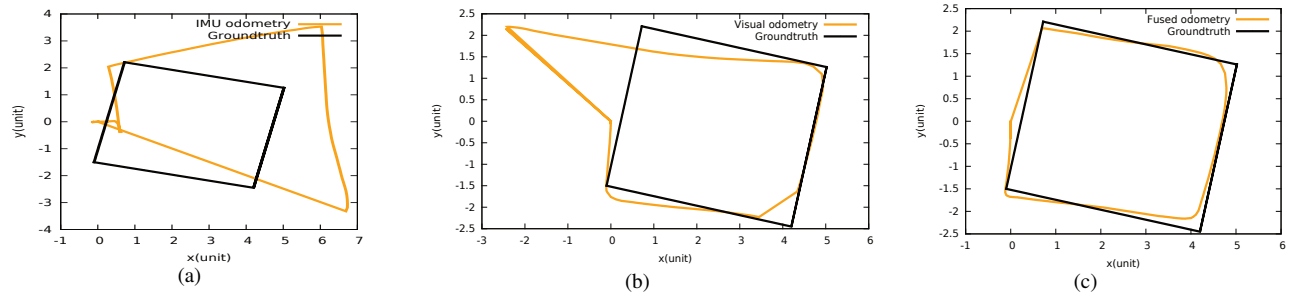

Figure 8: Comparison with approximate ground truth(GT). Orange curve depicts the output path and black curve is ground-truth. (a) shows output from IMU which is basically double integrated values of accelerometer; (b) shows output from visual odometry in comparison with the GT; (c) shows the output path after data fusion. The deviation from ideal path is reduced significantly after the fusion of visual odometry with the IMU data.

5 Results

5.1 Experimental Setup

In this example, we are showing results on *Google Glass Explorer 1* as proof of concept. GG runs Android Kitkat operating system (API 19). The device runs on Texas Instrument OMAP 4430 SoC, 1.2 GHz Dual ARM 7 processor. The device WiFi standard is 802.11b/g. For this experiment the GG captures the image frames in resolution 320×240. The specifications of back-end are 2.6 GHz Intel dual core i5-3320M Dell E6330 laptop. It has RAM capacity of 4GB and runs Ubuntu 14.04 operating system.

The application is tested at the office workplace in our research facility. The users were asked to wear the AR device and traverse (A1 to B1), (A2 to B2) and complete two loops of R.

5.2 Discussion

The application generates the topological map of the environment. Hence, it cannot be directly compared with the actual path taken by the user. Refer Figure 7 for comparing ideal path to be traversed against actual reported trajectory using bundle adjustment and without using it. All the map overlay reported use sensor fusion. The resultant map shows a convincing similarity with ideal paths due to bundle adjustments (Refer Fig. 7 (d) and 7 (f)). There are no loop closure detected on path A1-B1, hence Fig 7(b) is the final result on this path.

In another experiment, Figure 8 compares the deviation from ground-truth map of R. The calculated dimension of ideal path of R is 4.4 x 3.8 m^2. Having the simplicity in this path, the topological map output of R path is scaled and shifted to match it with ideal path and study the behavior from individual data source and post fusion output. This time user was asked to traverse only one loop of R. All the map overlay reported here use bundle adjustments where the end point meets starting point after loop closure detection. As we can infer from the result, clearly the map deviates due to noise and drift if we use accelerometer and visual odometry alone. The fused data outputs a much convincing map overlay. It is confirmed that, combining information from two different distribution of same phenomenon lead to a better estimation. Figure 9 shows the the

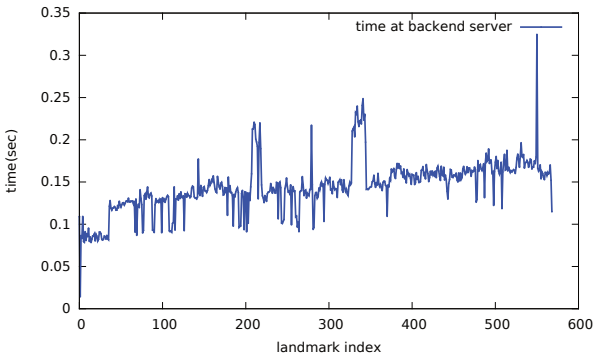

Figure 9: Time taken to report a location index by localisation method at back-end server. On an average location was reported in 0.15 seconds.

variation of time taken to report the current landmark at the back-end. The average time taken for total processing at back-end is 0.15 seconds. On the AR device map overlay consumes average 0.03 sec for plotting. Also the average total time taken by data exchange between AR device and back-end is 0.019 seconds. Here the processing at back-end determines the performance of the system. To sum up, the wearable AR device takes 0.2 seconds on average to append the map overlay with new node location.

The results are quite encouraging as the GG performs at 5fps and generated topological maps are similar to anticipated user's path.

6 Conclusions and Future Work

We have presented a framework for localisation and navigation that runs on the wearable AR devices. The novel features are two-fold: First, the EKF based fusion is performed on *Visual odometry* and *Accelerometer sensor data* for robust and dynamic topological map generation with bundle adjustments. We have utilized Android device orientation sensor in addition to odometry for pose estimation; Second, the bottleneck of limited processing capability of the GG reported for real-time applications has been addressed efficiently in our paper with the demonstration of a working prototype. In the future, we would incorporate the contextual information for improved navigation feature in our prototype.

References

[1] G. Bleser and D. Stricker. Advanced tracking through efficient image processing and visual–inertial sensor fusion. *Computers & Graphics*, 33(1):59–72, 2009.

[2] A. Bruhn, J. Weickert, and C. Schnörr. Lucas/kanade meets horn/schunck: Combining local and global optic flow methods. *International Journal of Computer Vision*, 61(3):211–231, 2005.

[3] P. Corke, J. Lobo, and J. Dias. An introduction to inertial and visual sensing. *The International Journal of Robotics Research*, 26(6):519–535, 2007.

[4] A. Developers. Available at `http://developer.android.com/reference/android/hardware/Sensor.html#TYPE_LINEAR_ACCELERATION`. Website, 2016.

[5] A. Developers. Available at `http://developer.android.com/reference/android/hardware/Sensor.html#TYPE_ORIENTATION`. Website, 2016.

[6] GoogleTechTalks. Available at `https://www.youtube.com/watch?v=C7JQ7Rpwn2k`. Website, 2016.

[7] J.-B. Hayet, F. Lerasle, and M. Devy. A visual landmark framework for indoor mobile robot navigation. In *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, volume 4, pages 3942–3947. IEEE, 2002.

[8] J. D. Hol, T. B. Schön, H. Luinge, P. J. Slycke, and F. Gustafsson. Robust real-time tracking by fusing measurements from inertial and vision sensors. *Journal of Real-Time Image Processing*, 2(2-3):149–160, 2007.

[9] S. Ishimaru, K. Kunze, K. Kise, J. Weppner, A. Dengel, P. Lukowicz, and A. Bulling. In the blink of an eye: combining head motion and eye blink frequency for activity recognition with google glass. In *Proceedings of the 5th Augmented Human International Conference*, page 15. ACM, 2014.

[10] N. Kejriwal, S. Kumar, and T. Shibata. High performance loop closure detection using bag of word pairs. *Robot. Auton. Syst.*, 77(C):55–65, Mar. 2016.

[11] F. Li, C. Zhao, G. Ding, J. Gong, C. Liu, and F. Zhao. A reliable and accurate indoor localization method using phone inertial sensors. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, pages 421–430. ACM, 2012.

[12] J. Z. Liang, N. Corso, E. Turner, and A. Zakhor. Image based localization in indoor environments. In *Computing for Geospatial Research and Application (COM. Geo), 2013 Fourth International Conference on*, pages 70–75. IEEE, 2013.

[13] T. Liu, M. Carlberg, G. Chen, J. Chen, J. Kua, and A. Zakhor. Indoor localization and visualization using a human-operated backpack system. In *Indoor Positioning and Indoor Navigation (IPIN), 2010 International Conference on*, pages 1–10. IEEE, 2010.

[14] J. Lobo and J. Dias. Relative pose calibration between visual and inertial sensors. *The International Journal of Robotics Research*, 26(6):561–575, 2007.

[15] S. M. Lowry, G. F. Wyeth, and M. J. Milford. Towards training-free appearance-based localization: probabilistic models for whole-image descriptors. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 711–717. IEEE, 2014.

[16] Z. Lv, L. Feng, H. Li, and S. Feng. Hand-free motion interaction on google glass. In *SIGGRAPH Asia 2014 Mobile Graphics and Interactive Applications*, SA '14, pages 21:1–21:1, New York, NY, USA, 2014. ACM.

[17] B. Mandal, S.-C. Chia, L. Li, V. Chandrasekhar, C. Tan, and J.-H. Lim. A wearable face recognition system on google glass for assisting social interactions. In *Computer Vision-ACCV 2014 Workshops*, pages 419–433. Springer, 2014.

[18] E. Martin, O. Vinyals, G. Friedland, and R. Bajcsy. Precise indoor localization using smart phones. In *Proceedings of the international conference on Multimedia*, pages 787–790. ACM, 2010.

[19] M. Milford, W. Scheirer, E. Vig, A. Glover, O. Baumann, J. Mattingley, and D. Cox. Condition-invariant, top-down visual place recognition. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 5571–5577. IEEE, 2014.

[20] N. Ravi, P. Shankar, A. Frankel, A. Elgammal, and L. Iftode. Indoor localization using camera phones. In *Mobile Computing Systems and Applications, 2006. WMCSA'06. Proceedings. 7th IEEE Workshop on*, pages 49–49. IEEE, 2006.

[21] J. Shi and C. Tomasi. Good features to track. In *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*, pages 593–600. IEEE, 1994.

[22] O. Woodman and R. Harle. Pedestrian localisation for indoor environments. In *Proceedings of the 10th International Conference on Ubiquitous Computing*, UbiComp '08, pages 114–123, New York, NY, USA, 2008. ACM.

[23] O. J. Woodman. An introduction to inertial navigation. *University of Cambridge, Computer Laboratory, Tech. Rep. UCAMCL-TR-696*, 14:15, 2007.

View publication stats