Evaluation of Different SLAM Algorithms using Google Tangle Data

Liyang Liu, Youbing Wang, Liang Zhao, Shoudong Huang
Centre for Autonomous Systems, Faculty of Engineering and IT, University of Technology, Sydney, Australia
{LiYang.Liu,Youbing.Wang,Liang.Zhao,Shoudong.Huang}@uts.edu.au

***Abstract*—In this paper, we evaluate three state-of-the-art Simultaneous Localization and Mapping (SLAM) methods using data extracted from a state-of-the-art device for indoor navigation — the Google Tango tablet. The SLAM algorithms we investigated include Preintegration Visual Inertial Navigation System (VINS), ParallaxBA and ORB-SLAM. We first describe the detailed process of obtaining synchronized IMU and image data from the Google Tango device, then we present some of the SLAM results obtained using the three different SLAM algorithms, all with the datasets collected from Tango. These SLAM results are compared with that obtained from Tango's in-built motion tracking system. The advantages and failure modes of the different SLAM algorithms are analysed and illustrated thereafter. The evaluation results presented in this paper are expected to provide some guidance on further development of more robust SLAM algorithms for robotic applications.**

I. Introduction

Simultaneous Localization And Mapping (SLAM) is a popular yet challenging research frontier in the robotic community. Many research efforts aim at producing accurate, robust and real-time SLAM algorithms. This, combined with availability of affordable devices and sensing platforms, brings a fully automated robot close to reality [1]. Feature-based SLAM is one of the most studied areas due to its usage of low cost camera sensor and ability to produce accurate results. Visual Inertial Navigation System (VINS) provides scale information fusing inertial measurements with vision input. Preintegration VINS has greatly enhanced VINS by turning it into a practical and high-fidelity solution [2][5][6]. Bundle Adjustment (BA) is known to provide accurate estimates of camera localizations as well as sparse scene reconstruction [8]. Parallax angle based Bundle Adjustment [3] offers a robust BA solution that handles even very challenging motion and scene situations. ORB-SLAM [4], with a modular design approach, turns the computation intensive optimization problem into a fully online real-time application. Leveraging on existing research achievements, Google has produced an engineering device *Tango* that detects its position relative to the surrounding world without using GPS [10].

Each of these methods has its own selling point, yet exhibit issues that cannot be overlooked. A SLAM researcher needs to evaluate these methods in a diverse environment and motion conditions to obtain good insight in the topic, with the help of a consistent and reliable data source.

In this paper, we explain a software solution to stream synchronized visual and inertial data from the Tango device into ROS – the widely-used robotic software development platform. With the same input data feed, we compare the aforementioned SLAM methods and outline our findings.

This paper is organized as follows. Section II introduces how we achieved data streaming from Tango. Section III describes the three algorithms that will be compared with the Google Tango trajectory results. Then, we provide the detailed analysis of Tango data and the experimental comparison results in Section IV. Finally, we conclude our analysis with future work proposed in Section V.

II. Obtaining Data from Google Tangle

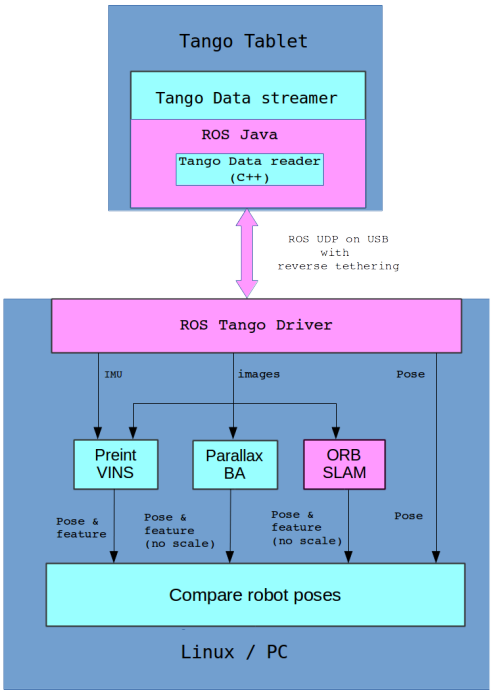


Fig. 1: Tango streaming system architecture

Google Tango is a technology platform that uses computer vision to track its position relative to the surrounding world [10]. The tango device model used in this research is a `Yellowstone` tablet. It features an RGB-D sensor, an IMU and a fish-eye camera. We developed a data streaming system for extracting data from the Tango and feeding it into

a Linux PC running ROS. Communication between tablet and PC is channelled via the USB network interface using reverse-tethering technology [13]. Our software package is essentially a two-driver system: a Tango data streamer running on Android encompassing the ROS-Java framework; and a Linux-side ROS driver node that decodes incoming data and broadcasts repackaged Tango messages into Linux ROS space. Within ROS, live Tango messages can be processed easily by many popular ROS modules, e.g. ORB-SLAM for on-line processing or `image_saver` for still image extraction and off-line analysis. We also used in-house developed MATLAB modules: Preintegration VINS and ParallaxBA to process the saved images off-line for algorithm comparison. The software architecture design is shown in Figure 1.

Data fetching in Tango is implemented as a real-time event driven system relying on call-back functions. We developed Pose and Image call-backs following Tango API C++ [11], and IMU call-backs based on Android API Java [12]. Tango/PC networking is implemented via the ROS-Java library. These fixed-time callbacks all provide mechanisms for fetching Tango generated sample timestamps. This implies all sampling event times (including poses, IMU and images), are synchronized on the same Tango clock. This view is confirmed in time analysis experiments detailed in Section IV-A1. We package all Tango timestamps into ROS messages for subsequent Linux domain processing. This treatment allows a fair algorithm comparison using precise timestamp correlation.

III. Algorithms for comparison

We briefly state the three SLAM algorithms that will be compared using Google Tango data, outlining their rationale and compelling features.

VINS is a system that fuses visual and inertial information from low cost camera and IMU devices and provide a trajectory with accurate scale information. The IMU sensor provides a data stream several orders of magnitude faster than the image rate, making a naive fusion of IMU and visual observations impractical to realize in real-time. Preintegration VINS (abbreviated Preint-VINS throughout this paper), proposed by Lupton and Sukkarieh [2] provides a convincing solution for real-time usage. The Preint-VINS algorithm combines many IMU samples as a single observation before fusion with camera images, resulting in a much reduced state and observation dimension in the optimization step. Further, inertial mathematical integration is performed in the robot body frame: robot state with respect to this frame is observable, hence its prediction uncertainty can be accurately obtained through linearization. In this new parameterization, the integrated inertial term (inertial delta) and feature pixels are treated as the observations, their uncertainties are used as weighting factors in optimizing for robot states and feature positions. The robot initial conditions, such as initial velocity, gravity and IMU bias, according to [2], can be automatically recovered in a linear manner.

The Preint-VINS algorithm proposed in [2] has been improved by using manifold representation [5] and continuous-time integration [6]. However for the comparison performed in this paper, we chose the Preint-VINS algorithm in [2] because of: its relatively easy implementation; and the performance improvements from [5] and [6] are only significant when the rotation changes are close to the Euler angle singularity or the IMU rate is low (neither of which apply in our experiments). Evaluating better Preint-VINS algorithms such as those in [5] and [6] is left as future research work.

A. ParallaxBA

BA is regarded as the gold standard for structure from motion. By minimizing the reprojection error, BA optimizes the feature positions and camera poses up to a scale [7]. The particular BA method we focus on is ParallaxBA – a Bundle Adjustment method proposed by Zhao et al. [3] that uses parallax angles for feature parametrization. Such a parameterization is more consistent with the projective nature of image formation therefore it: successfully avoids many singularity issues often found in along camera axis motion with traditional XYZ parameterization; offers faster convergence rate; and outputs more accurate motion and structure estimates [3]. Our experimental results confirm this claim. The source code of ParallaxBA back-end can be found on OpenSLAM website [1].

B. ORB-SLAM

ORB-SLAM is a real-time feature-based SLAM system. In essence, it is a carefully orchestrated framework that splits a traditionally computation intensive BA system into a few inter-dependent parallel modules for the purpose of real-time application. It includes: fast ORB feature extraction (viewpoint and illumination invariant); a stringent initialization stage of a reliable map that enables accurate feature tracking (avoids brute-force correspondence formation); on-going map update with new points and poses that are optimized in local region BA only; and efficient loop closure detection and global pose graph optimizaiton on essential nodes only. The ORB ecosystem coherently works together to produce robot trajectory and mapping, yet avoiding excessive computation seen in full BA whenever possible.

ORB-SLAM source code is available at ORB-SLAM project webpage[2] and is used in this work.

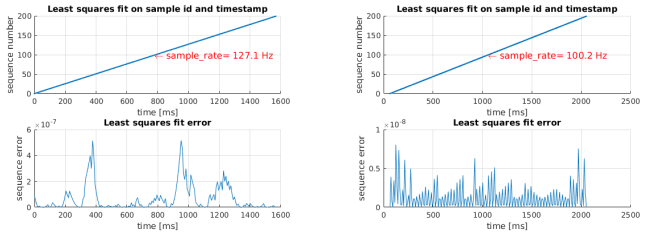
IV. Experiments

A. Tango data analysis

1) Tango data time synchronization: To evaluate Tango performance, we have performed extensive time sample analysis on collected IMU and pose samples. Applying least squares fit on sample times and sample sequence number, we saw negligible fit errors, therefore conclude the periodic nature of Tango data sampling. Also from least squares fit, we uncovered Tango gyroscope and accelerometer sampling rates to be 100 and 127 Hz respectively, as shown in Figure 2. Using these rates, we are able to detect missing samples in

[1] https://openslam.org/ParallaxBA.html
[2] https://github.com/raulmur/ORB_SLAM

long-term streaming and regenerate the missing ones. Preint-VINS requires unified accelerometer and gyroscope sample times, we therefore converted accelerometer data to match with gyroscope's sample times by linear interpolation. Once live data is available on ROS, we then visually inspected IMU and pose samples under various motion conditions. We are able to observe that they exhibit matched time responses to motion and conclude that they are well correlated in time. Also through visual inspection, we observed that the Tango reported pose values (translation and orientation) accurately reflect device motion. These investigations confirm our view that Tango is a reliable VINS data and Ground Truth provider, it is beneficial to VINS researchers as we can avoid the nasty engineering task of clock synchronization if a separate camera, IMU and Ground Truth provider were used.



(a) (b)

Fig. 2: Least Squares fit on sample time and sequence number reveals periodic sampling nature and rate: *a*) Accelerometer, *b*) Angular velocity.

2) Tango accelerometer bias analysis: Accurate estimation of Tango IMU bias is vital to success in computing trajectory. We used an empirical method to obtain good initial guesses for the accelerometer bias. As shown in Figure 3, the tablet is laid still on a flat surface. The gravity vector under such an orientation should be [0, 0, -9.8] m/s^2. The accelerometer reading acc_{flat} should be:

$$acc_{flat} = -g_{flat} + bias_f \tag{1}$$

Taking away g, the left over is $bias$ (assuming IMU is embedded flat inside Tango). Any further deviation from this initial guess should be very small which, according to [5]'s Monte Carlo analysis, should follow a linear relationship to the actual inertial delta measurements. Therefore, the final bias estimate can be safely obtained.

3) Tango gyroscope bias analysis: Tango's gyroscope exhibits excellent properties: we found it low in noise and have close-to zero bias and slow to drift. Integrating the gyroscope alone gives accurate measurement for device rotation: this rotation matches well with Tango's own reported orientation, also matches rotation computed from epipolar geometry between image frames.

4) Tango cameras: At the time of writing this paper, we are only able to decode Tango RGB camera images. The RGB camera has a small field of view: 62° x 37°, is subject to auto-white balancing and motion blur. Tango's own motion tracking system uses the Fisheye camera, it comes with global shutter

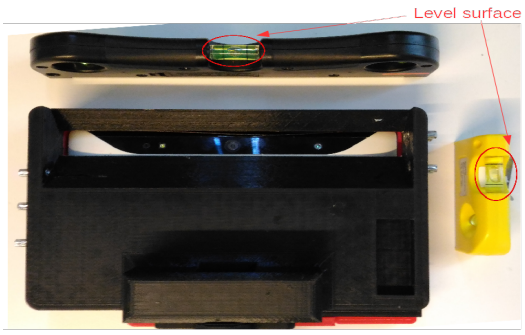


Fig. 3: Tango laid on a flat table to measure accelerometer bias

and 120° horizontal field of view, the richer feature resource puts Tango motion-tracking function in a more advantaged position than our own experiments using the RGB camera.

5) Tango calibration: Thanks to Google's documentation, all of Tango's calibration settings, including those of the two cameras (matrix and distortion coefficients), IMU noise level, and IMU and camera transformations can be readily read from the device.

B. Preint-VINS experimental setup

Motion with or without rotation can affect the VINS model. We noticed that in straight line motion (rotation free), gravity g and accelerometer bias $bias_f$, being in the same reference frame, are not separable. The inertial delta observation model [2] becomes degenerate. Only when a rotational motion is experienced, g and $bias_f$ can be separated through optimization.

In [2], Lupton was able to obtain good initial guesses of gravity and starting velocity, using first 3 observations from a stereo system. We are unable to modify Tango as a stereo system, therefore had to obtain initial conditions differently. Our solution is to start Tango in a stationary state, i.e. zero velocity, similar to Equation (1), Tango's accelerometer measurement should be the sum of $bias_f$ and g, subtracting the bias measured in Section IV-A2, the result gives g in the initial body frame.

Our in-house developed Preint-VINS module uses incremental optimization, it outputs full SLAM solution at end of processing. The initial stage performs batch processing on a small data segment, generating a good initial estimate for g and $bias_f$, then performs optimization on each incoming frame.

C. ORB-SLAM initialization

ORB-SLAM requires observation of significant parallax in order to triangulate for a set of initial map points reliable enough for subsequent point tracking. It takes indefinite time to start up, this drawback has been raised in [9] and encountered by us frequently. Our solution is to start data collection with a rotating motion around a scene in the center, giving both Preint-VINS and ORB-SLAM a good chance to kick start, then continue with random motion.

D. Compare results of four methods on same Tango data

For most experiments, we collected test data from a lab of 3m x 8m space, as shown in Figure 4(*b*). The only exception is the fast rotation test which took place on a flat table facing a scene hanging from the ceiling, shown in Figure 4(*a*).

(a) (b)

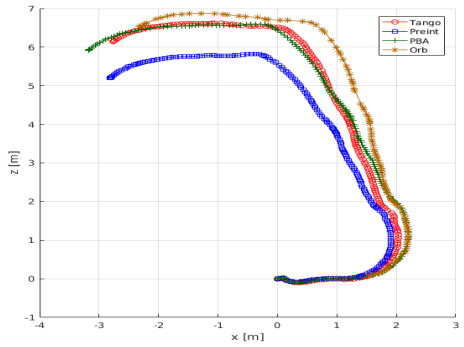
Fig. 4: *a*) Lab scene for long data collection. *b*) Fast rotation test scene: A planar ceiling as viewed by Tango laying on a flat table.

1) All methods working: The first test case we present here is that all four methods successfully produced results. Tango, Preint-VINS and ParallaxBA are able to start from the beginning and work till the end. ORB-SLAM took a short time to initialize. This is why ORB-SLAM trajectory appears later than others and not from location [0, 0, 0]. Moreover, ParallaxBA and ORB-SLAM do not produce scaling information. Since there's no Ground Truth in this test, we used Tango then Preint-VINS respectively to provide scaling for comparison, as shown in Figure 5.

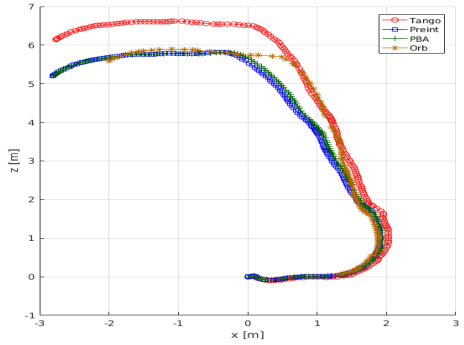
In both cases, ParallaxBA's trajectory follows closely with the scale provider. ORB-SLAM shows obvious deviation from other methods during a few segments. This is conceivable as ORB-SLAM only performs BA in the current frame's local neighbourhood [4]. In the absence of loop closure, ORB-SLAM will not recover from this error. Figure 6 shows a detailed comparison on orientation. ParallaxBA has smallest orientation error compared to other methods, ORB-SLAM again has most erroneous results.

The Preint-VINS trajectory has a smaller scale compared to Tango, this could be due to that our VINS implementation enforces constant bias. The system would strive very hard working out an optimal value for the entire 60 seconds of data, some compromise would have been made. Despite this, Preint-VINS is the only method that produced properly scaled trajectory and map (shown in Figure 7), Google Tango does not produce a map.

2) Tango and Preint-VINS work, ParallaxBA and ORB-SLAM fail: In Figure 8, we present the test case where the device is subject to fast small radius rotation over a duration of 3 seconds, facing a largely planar scene. Preint-VINS converged gracefully, producing a trajectory closest to Tango, fitting well with the experimenter's observation. ParallaxBA, although was able to converge, gave erroneous results. ORB-SLAM completely failed to kick-start. ParallaxBA builds its initial guess using Epipolar Geometry. If the scene is planar, finding the fundamental matrix is not well constrained [7][4], attempting to recover the motion from the fundamental matrix would produce wrong results. ORB-SLAM's failure could be explained by low feature match under fast rotation conditions.



(a) Using Tango scale, PBA very similar to Tango



(b) Using Preint-VINS scale, PBA very smilar to Preint-VINS

Fig. 5: rajectory comparison in XOZ plane: Tango, PreintVINS, ParallaxBA, ORB-SLAM

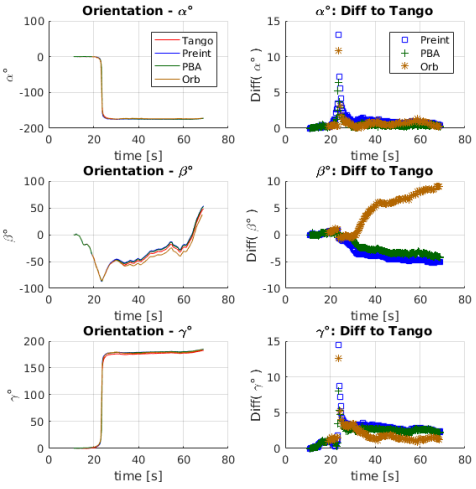


Fig. 6: Compare orientation (in degrees): Tango, Preint-VINS, ParallaxBA, ORB-SLAM

Preint-VINS worked well as anticipated, the IMU sensor provided necessary information overcoming poor observability of landmarks.

3) Tango, Preint-VINS and ParallaxBA work, ORB-SLAM fails: In Figure 9, ORB-SLAM completely fails to start. The

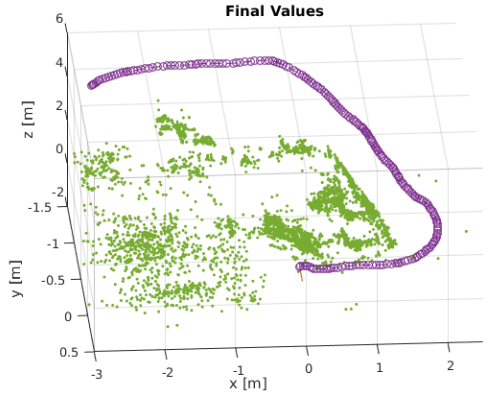


Fig. 7: Preint-VINS is the only method that provides properly scaled trajectory and map

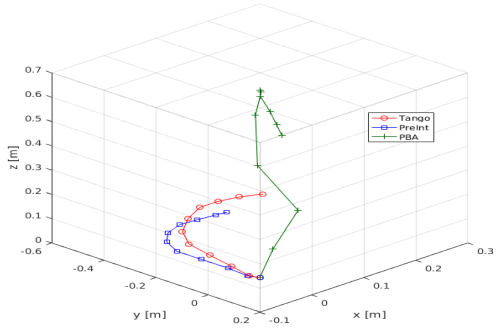
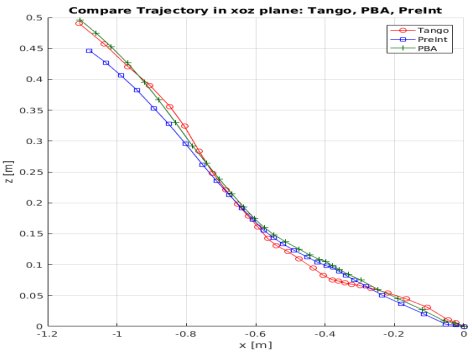
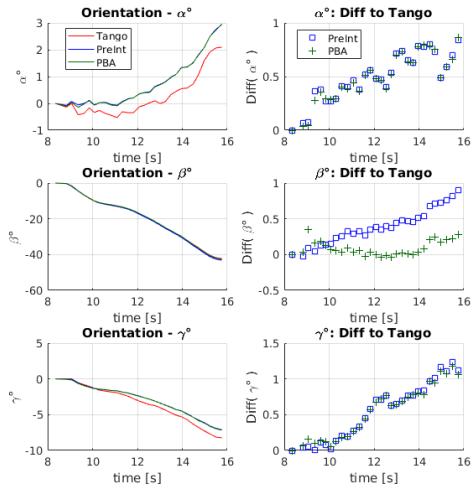


Fig. 8: Trajectory comparison: ParallaxBA shows largest error



(a) Trajectory comparison in XOZ plane (ParallaxBA use Tango scale)



(b) Orientation comparison (in degrees), all very similar

Fig. 9: Test case: Tango, Preint-VINS and ParallaxBA succeed, ORB-SLAM fails

scene tested here is feature-rich, ORB-SLAM's failure can only be explained by low parallax angle encountered between frames, see Section IV-C.

4) Tango, ParallaxBA and ORB-SLAM work, Preint-VINS fails: In Figure 10, Preint-VINS fails to find a convergent solution upon start-up. This may be overcome if we adopted an initialization stage similar to ORB-SLAM that continuously scans incoming data for convergence. However, that would imply that initial conditions such as gravity and velocity cannot be obtained in the manner described in Section IV-B. An alternative is to treat Tango as a stereo system, requires Fisheye images streaming, see Section IV-B.

E. Discussion

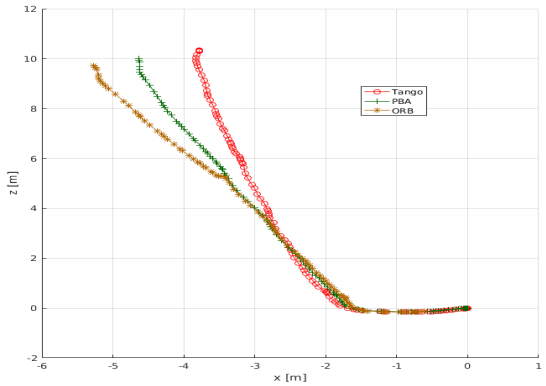
Our in-house developed Preint-VINS and ParallaxBA modules use same front-end for feature extraction, this front-end uses SIFT [16] for feature detection and RANSAC [17] for fundamental matrix calculation. Although such a method can reject many outliers, but is not completely effective.

ParallaxBA, due to its unique way of feature parameterization, is able to robustly handle outliers and challenging motion conditions. It is able to converge for almost all test cases.

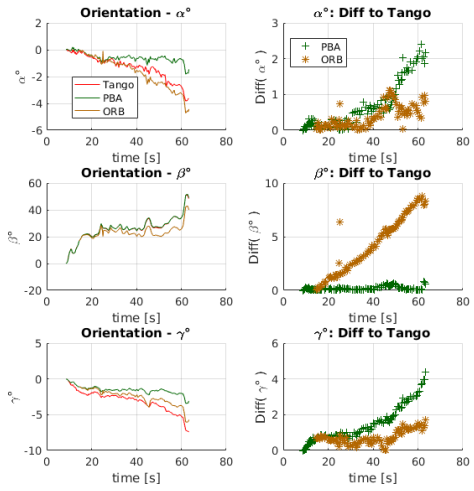
Our VINS system, currently using classic XYZ coordinates for feature parameterization, does not cope well with outliers or certain motion conditions. We choose to remove features showing small parallax angles to keep the system stable. Further, our implementation does not use a sliding window approach as [2], and cannot handle variable IMU bias yet. This makes the system prone to failure in mixed motion types (stationary + dynamic). Possibly because IMU does not restore to the same level when the device comes to rest. We have noticed that ORB-SLAM can start up later yet last longer than Preint-VINS.

ORB-SLAM requires certain motion conditions to initialize properly. We have noticed ORB-SLAM fail in many test scenarios due to inability to kick-start. Once initialized, ORB-SLAM uses feature tracking to localize, this has the advantage of fast data association in real-time application. However, track loss also happens due to swift camera motion.

Counting all tests performed, ParallaxBA is found to be the most robust method, it is always able to converge, results are in general accurate except for extreme feature-poor cases.



(a) Trajectory comparison in XOZ plane (ParallaxBA and ORB-SLAM use Tango scale)



(b) Orientation compairson (in degrees): ORB-SLAM shows largest deviation

Fig. 10: Test case: Tango, ParallaxBA and ORB-SLAM succeed, Preint-VINS fails

V. Conclusion and Future Work

In this paper, we compared three types of feature-based SLAM methods using data extracted from the Google Tango tablet. We verified Tango as a convenient data acquisition platform for producing synchronized IMU, visual and reliable robot trajectory information. Feeding the same Tango data to the three methods, we showed that Preintegration VINS can produce poses and map with correct scaling, while both ParallaxBA and ORB-SLAM can produce unscaled camera poses and feature locations. In terms of accuracy, Tango, Preint-VINS and ParallaxBA all produce reliable pose estimates, ORB-SLAM often deviates from the rest. In terms of computation time, Tango and ORB-SLAM both are functioning real-time solutions. ParallaxBA is most robust under challenging motion conditions. Preint-VINS can produce the most complete results: pose, feature, scale and real-time application if convergent conditions are met.

In comparing these algorithms, we treat Tango generated pose as quasi-Ground Truth, our visual inspection shows they do accurately reflect device motion. The three algorithms use image input from Tango's RGB-D sensor with long shutter speed and narrow FOV, different to Tango's internally accessed wide view images. To avoid the chance of motion blur, our data collections are of short-duration. Due to limitations in the test environment, they do not include complete loops either. As the next step, we will improve the current realization of Preint-VINS [6] to enhance its robustness and computational efficiency. We will also evaluate the algorithms further using larger scale and loopy datasets collected by Tango.

References

[1] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. D. Reid, and J. J. Leonard, "Simultaneous localization and mapping: Present, future, and the robust-perception age," *CoRR*, vol. abs/1606.05830, 2016. [Online]. Available: http://arxiv.org/abs/1606.05830

[2] T. Lupton and S. Sukkarieh, "Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 61–76, 2012.

[3] L. Zhao, S. Huang, Y. Sun, L. Yan, and G. Dissanayake, "ParallaxBA: bundle adjustment using parallax angle feature parametrization," *International Journal of Robotics Research*, vol. 34, no. 4-5, pp. 493–516, 2015.

[4] R. Mur-Artal, J. Montiel, and J. D. Tardós, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[5] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "IMU Preintegration on Manifold for Efficient Visual-Inertial Maximum-a-Posteriori Estimation," *Robotics: Science and Systems*, vol. 11, no. 6, July 2015.

[6] K. Eckenhoff, P. Geneva, and G. Huang, "High-Accuracy Preintegration for Visual Inertial Navigation," 2016. [Online]. Available: http://wafr2016.berkeley.edu/papers/WAFR_2016_paper_95.pdf

[7] R. Hartley and A. Zisserman, "Multiple View Geometry in Computer Vision," Cambridge University Press, second edition, 2004.

[8] B. Triggs, P.F. McLauchlan, R.I. Hartley, and A.W. Fitzgibbon, "Bundle AdjustmentA Modern Synthesis," in *Vision Algorithms: Theory and Practice*, pp.298–373, 2000

[9] S. Fujimoto, Z. Hu, R. Chapuis, and R. Aufrre, "ORB-SLAM map initialization improvement using depth," in *2016 IEEE International Conference on Image Processing (ICIP)*, pp. 261–265, Phoenix, USA, Septmber 2016.

[10] Wikipedia, "Tango," 2016. [Online]. Available: https://en.wikipedia.org/wiki/Tango_(platform).

[11] Google, "Tango C API," 2016. [Online]. Available: https://developers.google.com/tango/apis/c/.

[12] Google, "API Guides," 2016. [Online]. Available: https://developer.android.com/guide/index.html.

[13] C. Hoffman, "How to Connect Your Android to Your PCs Internet Connection Over USB," 2012. [Online]. Available: http://www.howtogeek.com/117118/how-to-connect-your-android-to-your-pcs-internet-connection-over-usb/.

[14] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," *2011 International conference on computer vision*. IEEE, pp. 2564–2571, Barcelona, Spain, November 2011.

[15] D. Gálvez-López and J. D. Tardos, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, 2012.

[16] D.G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[17] M.A. Fischler and R.C. Bolles, "Random sample consensus: A paradigm for model fitting with applicaitons to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[18] H. Strasdat, J. Montiel, and A. Davison, "Scale drift-aware large scale monocular SLAM," in *Robotics: Science and Systems (RSS)*, Zaragoza, Spain, June 2010.