IEICE TRANS. INF. & SYST., VOL.E101–D, NO.8 AUGUST 2018

LETTER

Robust 3D Surface Reconstruction in Real-Time with Localization Sensor

Wei LI[†a)], ***Nonmember***, **Yi WU**[†b)], ***Student Member***, **Chunlin SHEN**[†c)], ***and*** **Huajun GONG**[†d)], ***Nonmembers***

SUMMARY We present a system to improve the robustness of real-time 3D surface reconstruction by utilizing non-inertial localization sensor. Benefiting from such sensor, our easy-to-build system can effectively avoid tracking drift and lost comparing with conventional dense tracking and mapping systems. To best fusing the sensor, we first adopt a hand-eye calibration and performance analysis for our setup and then propose a novel optimization framework based on adaptive criterion function to improve the robustness as well as accuracy. We apply our system to several challenging reconstruction tasks, which show significant improvement in scanning robustness and reconstruction quality.

key words: *dense tracking and mapping, surface reconstruction, sensor fusion, RGB-D SLAM*

1. Introduction

3D reconstruction, especially dense surface reconstruction has gained significant interests in computer graphics, vision, and robotics communities. Especially with the eruption of VR/AR in recent years, dense model acquisition approaches or systems which are simple, robust and friendly to inexperienced users, are greatly demanded.

The real-time reconstruction systems, which are closely related to simultaneous localization and mapping (SLAM) techniques, mainly involve two aspects, the continuous camera tracking, and dense mapping. In this line of research, the representative work named DTAM [1], which could real-time model small scenes densely, is quite inspiring and promising for AR applications. But large number of obvious color features are required for credible camera tracking due to only monocular camera are used.

In the other hand, with the emergence of commodity RGB-D sensors such as the Microsoft Kinect and Asus Xtion, tracking and mapping with RGB-D sensors, which is also known as RGB-D SLAM [2], [3], opens new avenues for real-time dense reconstruction. By leveraging depth data from the RGB-D sensor, tracking becomes computationally cheaper and faster. Additionally, with the advance of general purpose graphics processing units (GPGPU), those methods can even produce decent models for large-scale scenes instantly. Regarding camera tracking component, most of those depth based methods use a variant of the iterative closest point (ICP) [4] algorithm to align input depth data. Although the ICP variants could provide dependable tracking results in some scenarios, keeping all tracking out of the drift and lost is still challenging, especially when the scanning scenes laking sufficient geometric details or encountering large frame-to-frame motion. Thus, 3D scanning applications such as KinectFusion [2] are still very limited in practice and require carefully chose scanning trajectory.

To improve the robustness of camera tracking, variant methods based on extra sensors such as inertial measurement unit (IMU) [5]–[7] are explored. Integrating IMU into scanning systems is closely bounded up with sensor fusion topics. A conventional strategy is using Kalman filter related algorithms to fuse IMU and depth based tracking together [6]. Alternatively, pushing IMU output as an energy term into the tracking optimization function receives more attention recently, as it forms a more general and flexible framework. However, even kinds of optimization methods are proposed, IMU based tracking is still not perfectly settled. More specifically, IMU may relieve tracking from lost but has limited help on tracking drift, as sensor's translation is not directly measured but numerically computed via integral.

We present a very robust method to easily produce 3D scanned models in real-time. We utilize non-inertial localization sensor comes from HTC VR system [8], which makes our system naturally suitable for most AR/VR applications. The sensor's non-inertial localization property can play an important role in avoiding tracking drift and lost, which are the stubborn problems in conventional methods even using IMU. On the other hand, different with marker based visual tracking system, such sensor can output positions when visible to one or two calibration-free base stations. Thus, we can set up the system from scratch within several minutes, which makes it easy to be used for customized reconstruction tasks.

In this work, we first evaluate the property of such localization sensor for reconstruction usage. Experiments show that the sensor's output location and direction is not suitable for dense mapping due to sensor inaccuracy and unsynchronization in large motion. Therefore, we introduce an online pose refinement algorithm to achieve highly accurate tracking. The main contributions of our work are summarized as follows:

- We propose an alternative reconstruction system using

Manuscript received March 20, 2018.
Manuscript publicized May 14, 2018.
[†]The authors are with the Collage of Automation Engineering, Nanjing University of Aeronautics and Astronautics, China.
a) E-mail: liweimcc@163.com (Corresponding author)
b) E-mail: janeyi105@126.com
c) E-mail: clshenac@nuaa.edu.cn
d) E-mail: ghj301@nuaa.edu.cn
DOI: 10.1587/transinf.2018EDL8056

low-cost and easy-to-setup hardware. We apply the hand-eye calibration and detailed performance analysis for the sensor.
- We propose a novel adaptive weighting based online pose refinement to improve reconstruction accuracy while retaining robustness benefited from the localization sensor.
- We demonstrate our system in several cases, it works well even with challenging cases.

2. System

Our hardware system is shown in Fig. 1, which consists of two base stations and one localization sensor mounted on the top of a Microsoft Kinect camera. The system can be built from scratch easily in a couple of minutes by strapping base stations onto the tripods and power them up. Note that, the extrinsic calibration between the two base stations is not needed. Moreover, the system can even work with only one base station by just losing little precision.

2.1 Calibration

The localization sensor streams its pose $\mathcal{T}^s$ in the sensor's coordinate to the host computer via wireless at 90Hz. $\mathcal{T}^s$ is a 4×4 matrix indicates the sensor's absolute position and orientation. To integrate depth into a global model, the pose $\mathcal{T}_i^c$ of depth camera in the world coordinate could be computed by the following transformation:

$$\mathcal{T}_i^c = \mathcal{H}^{-1}\mathcal{T}_0^{s-1}\mathcal{T}_i^s\mathcal{H} \tag{1}$$

Where $\mathcal{T}_0^s$ and $\mathcal{T}_i^s$ are sensor's pose in sensor's coordinate of the initial frame and i^{th} frame, respectively. $\mathcal{T}_0^s$ varies with every scan task. Therefore, when starting a new scan, it is required to keep the sensor still for several seconds, then we would extract $\mathcal{T}_0^s$ as the median of first 50 frames. $\mathcal{H}$ is the matrix transform pose from sensor's coordinate to world coordinate. $\mathcal{H}$ indicates the relative pose between sensor's center and Kinect camera's center, which can be obtained using hand-eye calibration. Regarding calibration, we first capture dozens of IR images relative to a fixed check-board and record corresponding sensor's pose in the meantime, then extract depth camera's extrinsic matrices and compute $\mathcal{H}$ following method of [9].

2.2 Model Integration

At time frame i in scanning, with camera intrinsic parameters, we extract a vertex map $\mathcal{I}_i^D$ in camera's view from the input raw depth data after undistortion and unprojection. A corresponding normal map $\mathcal{I}_i^N$ is generated in the meantime using central difference. Similarly, we also generate an associated color map $\mathcal{I}_i^C$ from the input raw RGB data. Given the camera pose $\mathcal{T}_i$, the input maps can be transformed into the world coordinate then fused into the global scene model. Different from conventional volumetric representation and fusion, the global model we used is based on surfels following the method proposed by Keller [3]. Taken integrating the vertex map $\mathcal{I}_i^D$ as an example, every point in $\mathcal{I}_i^D$ can form a new surfel with position $p_j \in \mathbb{R}^3$ in world coordinate, which is transformed using camera pose $\mathcal{T}_i^c$. Then p_j will be integrated into global scene model as:

$$\bar{p}_j \leftarrow \frac{\bar{\omega}_j\bar{p}_j + \alpha p_j}{\bar{\omega}_j + \alpha}, \quad \bar{\omega}_j \leftarrow \bar{\omega}_j + \alpha \tag{2}$$

Here, $\bar{p}_j$ and $\bar{\omega}_j$ are associated surfel's position and weight in the previous fused global model using projective association [2]. The initial surfel's weight α is computed as $\alpha = e^{-\gamma^2/2\sigma^2}$, where, γ is the normalized radial distance of current depth to camera center, and $\sigma = 0.6$ is derived empirically. Similar to the position integrating, we also integrate surfel's normal and color into global model from the normal map $\mathcal{I}_i^N$ and the color map $\mathcal{I}_i^C$, respectively.

2.3 Sensor Performance Analysis

Different from the marker base tracking system, the pose streaming out from the localization sensor cannot be simply treated as tracking ground truth and directly used for global model integration due to factors such as large sensor noise, streaming delay and calibration inaccuracy. We experimentally evaluate the sensor's localization accuracy by scanning a desk corner. We compare reconstruction using sensor's output $\mathcal{T}^c$ for dense mapping directly with using ElasticFusion [3], which is the base framework of our

Fig. 1 Hardware system. Top row, the localization sensor (in red box) mounted on a Microsoft Kinect. Button row, two base stations.

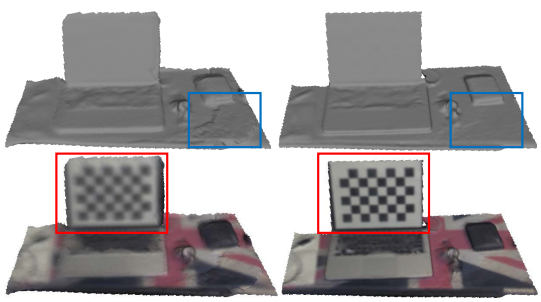

Fig. 2 From left to right, the results of using sensor's output pose for dense reconstruction and using ElastionFusion. The first row are the reconstructed models, while the second row are colorized using vertex color.

method. While scanning, we move camera within a small area. Figure 2 shows result using sensor's pose directly suffering from blurring and embossing. Actually, such problem can also be handled by extended Kalman filter (EKF), but EKF fails in other challenging cases shown in later experiments. Even though accuracy performance is unsatisfied, the sensor's non-inertial localization mechanism is still the attractive property we want to seize.

3. Online Pose Refinement

According to the sensor performance analysis in Sect. 2.3, the raw pose is not suitable for depth integration. Therefore, we propose a novel real-time pose refinement to improve pose accuracy while keeping sensor's ability to avoid drifting. We first explore four cues which will be used for the refinement.

3.1 Pose Metric

(1) Sensor and Velocity Metric

Although the pose from the sensor is not accurate enough, it is still the main guidance for truth and avoiding drifting. We affirm that camera pose should be close to observation, so we define a sensor metric in the energy form as:

$$E_{sen}(\mathcal{T}_i) = \|\mathcal{T}_i - \mathcal{T}_i^c\|_2 \tag{3}$$

Where $\mathcal{T}_i$ is final camera pose for dense integration. Next, we involve a velocity constraint to enforce camera's movement close to sensor's, this is defined in the form:

$$E_{vel}(\mathcal{T}_i) = \|\mathcal{T}_i\mathcal{T}_{i-1}^{-1} - \mathcal{T}_i^c{\mathcal{T}_{i-1}^c}^{-1}\|_2 \tag{4}$$

Where $\mathcal{T}_{i-1}$ and $\mathcal{T}_{i-1}^c$ are camera and sensor's pose of the previous frame, respectively. Those two constraints are simply derived from sensor's property. They can be used as the initial guess. Furthermore, they are essential for avoiding drift and improving robustness. However, they have a limited improvement in accuracy.

(2) Dense Registration Metric

In order to improve accuracy, we introduce two dense registration metrics, the geometric and photometric metric into our system. We compute the difference of input RGB-D frame with the rendered global model using target camera pose $\mathcal{T}_i$. Specifically, we render clipped surfels in the global model into 2D image plane using the surfel-splatting method to obtain vertex map $\mathcal{V}_i$, normal map $\mathcal{N}_i$ and color map C_i. Then, the geometric energy of frame i, which indicates the difference between points in input map $\mathcal{I}_i^D$ and rendered map $\mathcal{V}_i$, can be computed using the sum of point-to-plane distance:

$$E_{geo}(\mathcal{T}_i) = \frac{1}{n}\sum_j^n \Big((\mathcal{T}_i p_j - \mathcal{V}_i(\bar{p}_j)) \cdot \mathcal{N}_i(\bar{p}_j)\Big)^2 \tag{5}$$

Where p_j is the 3D point observed in the input vertex map $\mathcal{I}_i^D$, while $\bar{p}_j$ is the corresponding point in rendered vertex map associated by using projective data association. The photometric energy is formed by:

$$E_{pho}(\mathcal{T}_i) = \frac{1}{n}\sum_j^n \|I_i^C(\pi(T_i p_j)) - C_i(\bar{p}_j))\|_2 \tag{6}$$

Where I_i^C is the input color image, $\pi(p)$ denotes the perspective projection and dehomogenisation operation for a 3D point.

3.2 Energy Optimization

We utilize all metrics in Sect. 3.1 to compute accurate camera pose for dense mapping. However, simply combining all metrics together is problematic since different metrics have varying properties. Therefore, we adopt a scalable criterion function to adaptively adjust metric's weight. The total energy is defined as follows:

$$E_{total}^i = \sum_m \rho(E_m(T_i)/\beta_m) \tag{7}$$

Here, $\rho(r)$ is Cauchy's function defined as: $\rho(r) = q^2 ln(1+(r/q)^2)/2$, which is used to criticize gross energy errors and adjust each metric's weight separately, here parameter q is chosen to 4 empirically. While β_m is scale weight for each metric, which is used to tune the balance between metrics. According to [10], we can solve the following least square system to minimize the energy in Eq. (7):

$$T_i = argmin \sum_m \omega_m^2 E_m(T_i)^2/\beta_m^2 \tag{8}$$

Where $\omega(r) = 1/(1 + (r/q)^2)$ is named weight function [10] derived from $\rho(r)$. We approximate $\mathcal{T}_i \in \mathbb{R}^{4\times4}$ with a six-dimensional vector by linearizing the rotation, then minimize Eq. (8) within an iterative fashion. In each iteration, we update the point correspondence for geometric and photometric terms as well as the scale weight β_m. For sensor and velocity metric, they share the same iteration wise scale weight $\beta_1(k) = 1/k^2$, here $k \in \{1, 2, \ldots k_{max}\}$ is the current iteration index. While dense registration metrics share one scale weight $\beta_2(k) = 1 - 1/k^2$. The insight of iteration wise weight function β is temporally adjusting metric's importance. In the early stage, we tend to rely more on sensor's essential metrics to get good initial guess and limit search region. While in the later fine tune stage, dense registration metrics would take the more important role.

After convergence or reach maximum iteration limit, we get the total residual energy E_{total}^i, which we realize could indicate the frame's health weights for global integration. Therefore, we reuse it for adjusting the integrating weight for current frame. So, we rewrite α in Eq. (2) as $\alpha = e^{-r^2/2\sigma^2}/max(E_{total}, \bar{E})$, here $\bar{E}$ is low threshold bound used to avoid small energy E_{total}.

4. Experiments

We evaluate our system by reconstructing two scenes. Note

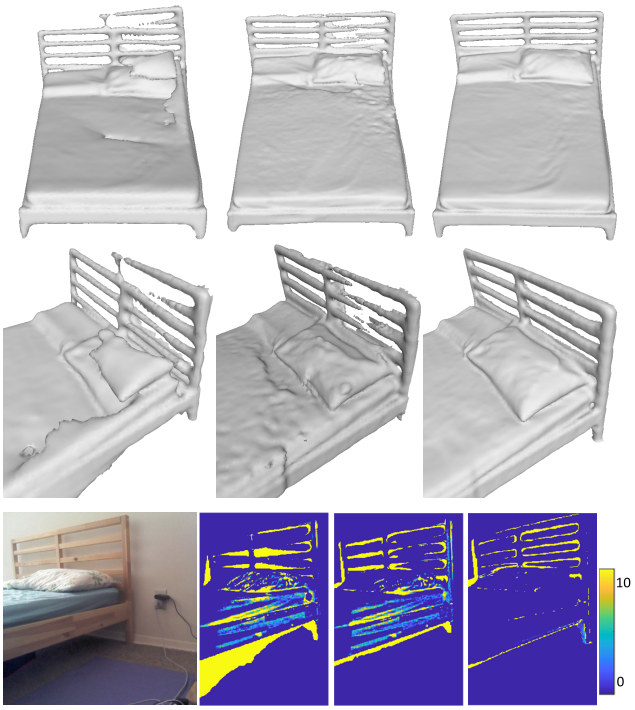

Fig. 3 The comparison of modeling a full-size bed using different methods. From left to right, results of using ElasticFusion [3], ElastionFusion with EKF and our method. Top two rows are results in front and close view, receptively. The third row, the input RGB image and its gray-scale difference with a rendered image from reconstructed models using those three methods.

that we record the scanning sequence for fair comparison even our system is designed to run online. All experiments are conducted using a laptop with quad-core CPU@i7-7700HQ, 8G RAM, GTX1060 6G GPU RAM.

We first evaluate our method by reconstructing a full-size bed. Results are shown in Fig. 3. The baseline is the state of art dense surface reconstruction method ElasticFusion [3]. We also implement an EKF based on ElasticFusion framework to utilize sensor. While scanning, we carry out some too close views to simulate arbitrary even ill-suited trajectory which always occurs while inexperienced users do scanning. ElasticFusion fails in the tracking of those extreme viewpoints. With sensor fusing using EKF, the result are better but still unacceptable, as EKF is a kind of combination of the sensor output and conventional tracking result. While, with the novel adaptive weighting refinement, our method generates the highest quality model.

Figure 4 shows comparison when modeling a more challenging scene, a green screen studio. As the walls are covered by the featureless plane green screens, this case cannot be handled by almost all of the state of the art methods using only RGB-D sensor when moving parallel to the screen. Obviously, ElasticFusion [3] gets poor result. While, with the localization sensor and our novel optimization framework, our system can still produce a high fidelity model in this challenging case.

A more detailed evaluation statistics are shown in Table 1 to quantitatively describe the improvement. We also carried out a comparison between using sensor's output direct for fusion (column 3) and using fixed metric weight optimization bypassing the adaptive weighting procedure in Sect. 3.2 (column 4). As lack of a ground true model, we compute the RMSE (root mean squared error) between the input image and the rendered image of the colorized model at the same viewpoint. Here, the RMSE is defined as: $\sqrt{\sum_k^n (g_k^i - g_k^r)^2 / n}$, where g_k^i and g_k^r are 8-bit gray-scale value ($0 \sim 255$) of k^{th} pixel in the input image and the rendered image, respectively. Note that, even slightly camera position drift or error, which always exist from the middle of scanning, would make the rendered and the input image with large pixel-wise match errors. Thus, we use only the first frame of the input sequence for comparison. As the Table 1 shown, our method has the smallest reconstruction error.

Fig. 4 The comparison of reconstructing a green screen studio. From left to right, results using ElasticFusion [3] and our method.

Table 1 Statistics of the reconstruction error of different methods using the RMSE of the input image and rendered image from the model.

Model	ElasticFusion	Sensor	Fixed Weight	EKF	Ours
Bed	8.35	6.92	6.46	6.33	**5.13**
Studio	6.16	5.09	5.03	4.95	**4.34**

5. Conclusion

We presented an easy-to-build setup and novel optimization framework to robust reconstruct 3D scenes in real-time. With the non-inertial localization sensor, our system can handle global drift and large motion naturally comparing with IMU based setups. The following novel adaptive weighting optimization is carried out to achieve high-quality result against inaccurate and noisy tracking result from the sensor. With our robust system, even inexperienced users can easily produce decent 3D models without thinking about proper scanning trajectory.

Acknowledgments

This work was supported in part by the National High-tech R&D Program (863 Program) of China under Grant 2015AA015905.

References

[1] R.A. Newcombe, S.J. Lovegrove, and A.J. Davison, "Dtam: Dense tracking and mapping in real-time," Computer Vision (ICCV), 2011 IEEE International Conference on, pp.2320–2327, IEEE, 2011.

[2] R.A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A.J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on, pp.127–136, IEEE, 2011.

[3] T. Whelan, S. Leutenegger, R.F. Salas-Moreno, B. Glocker, and A.J. Davison, "Elasticfusion: Dense slam without a pose graph," Robotics: Science and Systems, 2015.

[4] P.J. Besl and N.D. McKay, "A method for registration of 3-d shapes," IEEE Transactions on pattern analysis and machine intelligence, vol.14, no.2, pp.239–256, 1992.

[5] A.I. Mourikis and S.I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided inertial navigation," Robotics and automation, 2007 IEEE international conference on, pp.3565–3572, IEEE, 2007.

[6] J.C.L. Chow, D.D. Lichti, J.D. Hol, G. Bellusci, and H. Luinge, "Imu and multiple rgb-d camera fusion for assisting indoor stop-and-go 3d terrestrial laser scanning," Robotics, vol.3, no.3, pp.247–280, 2014.

[7] M. Nießner, A. Dai, and M. Fisher, "Combining inertial navigation and icp for real-time 3d surface reconstruction," Eurographics (Short Papers), pp.13–16, 2014.

[8] S. Buckley, "This is how valve's amazing lighthouse tracking technology works," Gizmodo. Retrieved, vol.2, 2015.

[9] C.X. Guo and S.I. Roumeliotis, "Imu-rgbd camera 3d pose estimation and extrinsic calibration: Observability analysis and consistency improvement," Robotics and Automation (ICRA), 2013 IEEE International Conference on, pp.2935–2942, IEEE, 2013.

[10] P.J. Huber, "Robust statistics," Wiley Series in Probability and Statistics, 1981.