Robust RGB-D Camera Tracking using Optimal Key-frame Selection

Kyung Min Han and Young J. Kim

Abstract— **We propose a novel RGB-D camera tracking system that robustly reconstructs hand-held RGB-D camera sequences. The robustness of our system is achieved by two independent features of our method: adaptive visual odometry (VO) and integer programming-based key-frame selection. Our VO method adaptively interpolates the camera motion results of the direct VO (DVO) and the iterative closed point (ICP) to yield more optimal results than existing methods such as Elastic-Fusion. Moreover, our key-frame selection method locates globally optimum key-frames using a comprehensive objective function in a deterministic manner rather than heuristic or experience-based rules that prior methods mostly rely on. As a result, our method can complete reconstruction even if the camera fails to be tracked due to discontinuous camera motions, such as kidnap events, when conventional systems need to backtrack the scene. We validated our tracking system on 25 TUM benchmark sequences against state-of-the-art works, such as ORBSLAM2, Elastic-Fusion, and DVO SLAM, and experimentally showed that our method has smaller and more robust camera trajectory errors than these systems.**

I. Introduction

Camera Tracking is a challenging research problem in robotics and computer vision communities. The emergence of the RGB-D camera made vision-based camera tracking problems even more prevalent. The depth sensor was originally invented for gaming devices; however, its application is now more diverse, and no longer limited to gaming devices. Because of the accurate depth information, in particular, 3D reconstruction results using depth sensors are much more robust and reliable than RGB imaging.

Numerous studies have been performed on RGB-D camera tracking [1], [2], [3], [4], [5]. The main structure of these systems is based on real-time VSLAM where camera poses and maps are constructed in response to the real-time camera motion, followed by pose optimization for loop-closing events. A popular approach to the VSLAM problem is the pose-graph formulation where an edge between two vertices represent a pose constraint including an odometry constraint between two ordinary frames and a loop closing constraint between two key-frames. Therefore, selecting good key-frames and a good initial camera pose impacts the final tracking result significantly.

Our goal in this research is proposing a robust RGB-D camera tracking system by addressing the following two questions:

- How to robustly perform RGB-D visual odometry (VO)?
- How to select good key-frames from a sequence of camera frames?

The authors are with the department of computer science and engineering at Ewha womans university in Korea {hankm|kimy}@ewha.ac.kr

Our answer to the first question is revisiting the hybrid VO method, which takes advantage of both structure and texture information in RGB-D scenes, and making it adaptive to the tracking environment. That is, our method autonomously determines the weighting factors to combine depth map and image texture information in an RGB-D image without human intervention.

In order to answer the second question, a novel integer programming based set covering scheme is proposed to robustly identify an optimal subset of frames to cover the entire image frames. Specifically, we construct an affinity matrix for the input frames by establishing *semi-dense* putative matches among them, where full key-point matches are not required. Then, the affinity relationship is fed back to the set covering scheme to find globally optimal key-frames. Note that many existing camera tracking systems rely on heuristic criteria or a locally optimum decision function to determine key-frames, but the accuracy and reliability of the reconstructed map is rather questionable.

To measure the robustness of our tracking system, we employed mean relative pose error (RPE) and mean absolute trajectory error (ATE) results over the 25 TUM benchmark sequences [6]. As shown in Section VI, our method exceeds the baseline methods in terms of these two metrics. In summary, the main contributions of our paper are:

- Adaptive VO method using a novel pose estimation formulation
- Novel integer programming (IP) based formulation for optimal key-frame selection
- More robust and accurate results compared to the state of the art VO and SLAM systems using public benchmark datasets

II. Related Work

An ordinary camera tracking system consists of front-end and back-end modules that are independent of each other. The front-end is responsible for estimating the camera trajectory by establishing the pixel correspondences between two consecutive frames, often referred to as visual odometry (VO). In parallel, the back-end of the system runs to correct the drift error possibly accumulated by the front-end of the system.

A. The Front-end

VO dictates the overall performance of a camera tracking system because a robust and accurate VO minimizes the drift error accumulation which can impact on the outcome of the entire system. VO is often categorized into two methods: direct methods and feature-based methods. The direct methods

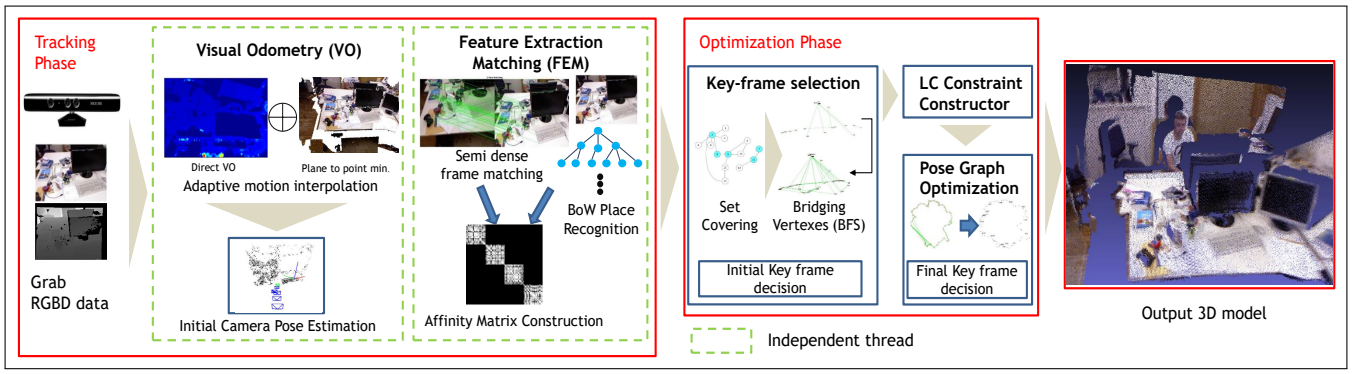


Fig. 1: Pipeline of the proposed tracking system. The tracking phase estimates incremental camera motions and simultaneously establishes similarities among the input frames in an affinity matrix. Then, during optimization phase, key-frames are identified followed by a pose graph optimization.

[2] establish dense correspondences by imposing a brightness consistency constraint. The camera pose is directly recovered by minimizing the pixel intensity error. Conversely, the feature-based methods [7], [4] locate salient pixels in the *image scale-space* followed by a matching process, such as a normalized cross correlation (NCC) measurement, the sum of squared differences (SSD), or descriptor distance. Imposing a camera epipolar constraint can further eliminate outlier matches in order to achieve more accurate VO estimation.

When the depth information is available as prior knowledge, we do not need to be confined in 2D-to-2D image alignment. Motion estimation from 3D-to-3D correspondence is feasible [8]. Furthermore, 3D point clouds can be directly registered using ICP-like methods [9]. Besides, there are RGB-D SLAM methods that rely on ICP registrations [10], [11].

B. The Back-end Optimizer

The importance of back-end module cannot be overstated because VO may accumulate the pose error for long-term operations. One way of correcting the pose error is frequently running non-linear optimization of cost functions built by relative camera pose constraints or by pixel reprojection error. The former refers to pose-graph-optimization [12], and the latter refers to bundle adjustment [13].

C. Global Relocalization

Camera tracking systems are equipped with an additional step for the global relocalization task. VO may lose tracking of the camera pose owing to numerous factors, such as insufficient feature matching or motion blur. Besides, the camera sensor could be kidnapped, and then released at a long base-lined location w.r.t. the kidnapped location. Furthermore, many loop closing constraints may exist among the key-frames with poses that must be corrected by a back-end process. While a bag of words (BoW)[14]-based approach is a well-known efficient method for the global camera localization, recent approaches [15], [16] are capable of coping with drastic appearance changes in scenes.

III. Overview

We use a sequential pipeline for our tracking system consisting of two consecutive phases: tracking phase (the front-end) and optimization phase (the back-end). The tracking phase estimates the primitive poses of input frames and their similarities while the optimization phase identifies a set of key-frames and then refines their poses. Figure 1 illustrates the proposed pipeline.

The next few sections elaborate on some crucial steps in the pipeline. Sections IV-A and IV-B explain VO to construct the affinity matrix and our feature matching method in the pipeline, respectively. Section V-A explains how key-frames are selected, and Section V-B describes our final pose optimization procedure.

IV. Tracking Phase

A. Visual Odometry

1) Cost Functions: The point to plane ICP residual ρ for the kth point $\mathbf{X}_k$ is defined as

$$\rho_k = (\mathbf{X}_k - \exp(\hat{\xi})\mathbf{C}\mathbf{X}'_k) \cdot \mathbf{n}_k \tag{1}$$

where $\hat{\xi} \in se(3)$ is the twist representing a rigid motion. $\mathbf{X}'_k$ and $\mathbf{n}_k$ are the corresponding 3D point and the normal vector, respectively. Then, the ICP cost function is given as:

$$E_i = \min_{\hat{\xi}} \sum_k \psi(\rho_k)||\rho_k||^2 \tag{2}$$

where $\psi(\cdot)$ is a robust weight function which minimizes the impact of false correspondences [17]. As for DVO, a pixel intensity residual δ is defined as

$$\delta_k = I_1(\mathbf{x}_k) - I_2(\mathbf{x}'_k) \tag{3}$$

where $\mathbf{x}'_k$ is the pixel correspondence of $\mathbf{x}_k$ computed by a warping operation; that is,

$$\mathbf{x}' = \pi(\mathbf{C}, \pi^{-1}(\mathbf{x}, \mathcal{Z}(\mathbf{x}))). \tag{4}$$

The resulting pixel intensity cost function is given as

$$E_d = \min_{\hat{\xi}} \sum_k \psi(\delta_k)||\delta_k||^2. \tag{5}$$

Solving (2) or (5) involves iteratively running the re-weighted least square (IRLS) algorithm. A method like [18] co-optimizes Eq. 2 and Eq. 5 using a pre-determined weight parameter to estimate the camera pose. Since it is known that ICP and DVO show different characteristic behavior depending on the availability of textures or depth structures in an input scenario [19], our algorithm leverages this fact by adaptively adjusting the weight parameter according to the relative fitness between DVO and ICP at each IRLS iteration. As a result, our VO algorithm can generate more robust and accurate results than [18].

2) Camera Pose Estimation: We downsampled images to four pyramid levels to implement coarse to fine search strategy; that is, beginning from the coarsest pyramid level (80×60 for a TUM sequence,) we iterate the IRLS to find the best camera pose at the current pyramid level. Then, we assess the quality of the estimated camera motions for n number of correspondences using

$$\begin{aligned} e_d &= \sum_{k}^{n} ||\mathbf{X}_k - \exp(\hat{\xi_d})\mathbf{X'}_k||_2 \\ e_i &= \sum_{k}^{n} ||\mathbf{X}_k - \exp(\hat{\xi_i})\mathbf{X'}_k||_2 \end{aligned} \tag{6}$$

where $\mathbf{X}_k$ and $\mathbf{X'}_k$ are the corresponding pair of two 3D points, $\hat{\xi_d}$ and $\hat{\xi_i}$ are the estimated camera twists from the IRLS of DVO and ICP, respectively. Here, the subscript d refers to DVO while i refers to ICP. Then, we introduce a gain parameter λ_d

$$\lambda_d = \begin{cases} \frac{e_i}{e_i + e_d} & \text{if } n \geq \nu \\ 0 & \text{if } n < \nu \end{cases} \tag{7}$$

where ν is the number of meaningful Sobel responses used for DVO process - we set ν to 2% of the image resolution. Given λ_d, $\lambda_i = 1 - \lambda_d$, we interpolate the two camera motions by

$$\mathbf{C}_{avg} = \mathcal{I}(\exp(\hat{\xi_i}), \exp(\hat{\xi_d}), \lambda_i, \lambda_d) \tag{8}$$

where $\mathcal{I}(\cdot)$ is a weighted interpolation function consisting of two consecutive steps: 1) orientation interpolation using quaternion Slerp [20] followed by 2) position interpolation in Euclidean space. The interpolated camera motion is used as an initial guess for the next pyramid level. A pseudo-code for our pose estimation procedure is also given in Algorithm 1.

B. Feature Extraction and Matching (FEM)

Semi-Dense Frame Matching: We build an affinity matrix by executing multiple wide baseline matchings for each incoming frame. To speed up building the affinity matrix, we performed a BoW match before putative matching to locate possible image pairs. The possible image pairs further verified by CudaSIFT [21] to complete the matching task. Here, we run the image/feature matching task as an independent thread in parallel to our VO thread to alleviate the time-consuming problem of this part of the system.

Function *RGBDODOM()*
  initialize the camera motion $\mathbf{C}$;
  for *each pyramid level* **do**
    for *each IRLS iteration* **do**
      solve Eq. 2 for ξ_d;
      solve Eq. 5 for ξ_i;
      recompute $\psi(\rho)$ and $\psi(\delta)$
    end
    compute λ_i and λ_d using Eq. 7 ;
    estimate $\mathbf{C}_{avg}$ using Eq. 8 ;
    $\mathbf{C} \leftarrow \mathbf{C}_{avg}$
  end

Algorithm 1: Camera pose estimation procedure

The result of η image matching is incorporated into the key-frame decision module in order to select optimal key-frames. The detailed explanation of the key-frame decision step is presented in Section V-A.

V. Optimization Phase

A. Key-frame Selection

In pose-graph SLAM, the unknown camera poses are considered a graph $G = (V, E)$ where a set of vertices V expresses camera poses created from its trajectory, and a set of edges E represents visual associations between the vertices in V. $v_i \sim v_j$ denotes that $v_i \in V$ and $v_j \in V$ are connected by an edge e_{ij}. In our work, we measured the number of feature matches between v_i, v_j to determine if they are connected.

Set Cover Formulation: Our goal is to select a minimal set of vertices V^* such that the selected vertices can cover the entire vertex set V. The selected key-vertices would cover their neighbors as well as themselves. A vertex v_k and its connected vertices form a sub-graph S_k of G. Therefore, multiple selections of key-vertices or sub-graphs would eventually cover the universe - i.e. the entire set of frames $\mathbb{U}$.

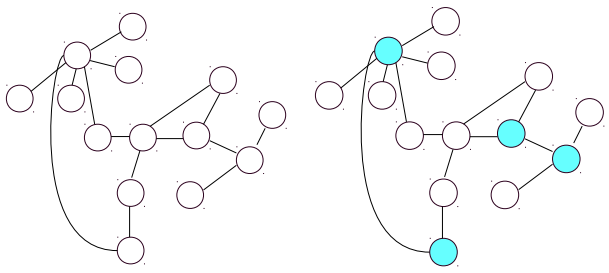
$$\bigcup_{V_i \in V^*} V_i = \mathbb{U}. \tag{9}$$

The formulation above is a set covering problem [22]. There are two main objectives for this formulation. First, we want to cover all vertices by key vertices. Second, a selected key vertex $v_i \in V^*$ should be connected to another key vertex $v_j \in V^*$, $i \neq j$ for pose graph optimization (PGO) step.

The weighted selection of the vertices should be optimized by minimizing a cost function:

$$\begin{aligned} \min_{x} \quad & \sum_{x_j \in V} w_j \cdot x_j \\ \text{s.t.} \quad & \sum_{j}^{N} a_{ij} x_j \geq 1 \quad i, j = 1...N \\ & \sum_{k} x_k - x_i \geq 0 \quad k \neq i, V_k \in S_i \\ & \sum_{k} x_k \geq c, \end{aligned} \tag{10}$$

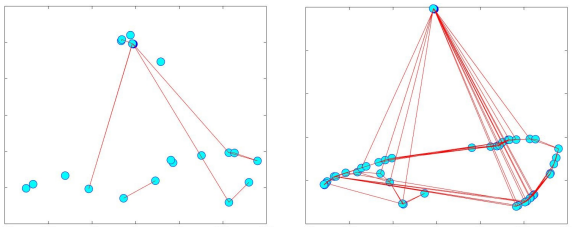
where the cost $w_j = 1/|S_j|$ is the inverse cardinality of a subgroup belonging to v_j. The decision variable x_j is set to one if v_j is selected; otherwise zero. The binary coefficient a_{ij} is set to one iff j^{th} vertex appears in the i^{th} group. The first constraint ensures the constraint in Eq. 9, and the second constraint avoids selecting a single isolated vertex. The optional third constraint guarantees to select at least c number of vertices. Figure 2 illustrates the proposed vertex cover process for a simple example



(a) A graph (b) Key-vertices

Fig. 2: The nodes in cyan color are the key-vertices found by our algorithm. Note that every vertex is covered by the key vertices.

1) Locating Bridging Vertices: Although the vertices selected from Eq. 10 cover the entire vertices, we observed a case where our algorithm finds multiple disconnected sets. In order to guarantee a single connected component of G, it is necessary to locate additional bridging vertices. We employed breadth first search (BFS) to locate the bridging frames. Note that the set of selected vertices from Eq. 10 together with the identified bridging vertices correspond to *key-frames*. Figure 3 shows key-frames search results of a real image data set. Note that all key-frames are formed as a single connected component.



(a) Without bridging frames (b) With bridging frames

Fig. 3: The key-frames selection result for TUM *fr1 desk2* sequence: (a) key-vertices found by optimizing Eq. 10 and (b) after locating bridging frames by BFS. Note that the BFS connects all key-frames so that all vertices form a connected component.

B. Pose Optimization

1) Initializing and Optimizing Loop Closing Constraints: Thanks to the affinity matrix constructed in the tracking phase, we can locate candidates of loop-closing view-pairs. We use Arun's [8] 3D to 3D registration method with a special treatment for the reflection case suggested by [23]. Given a 3D point pair, we initially estimate the relative pose constraint from a standard RANSAC protocol. In order to estimate accurate pose constraints, we further optimize the initial estimations by executing motion-only bundle adjustment followed by a median absolute deviation (MAD) based outlier elimination step. We first compute σ of MAD by

$$\sigma = 1.4826 r_i, \tag{11}$$

where r_i is the absolute deviation of the reprojection error $\mathbf{v}$ from its median residual - i.e. $r_i = |\,\|\mathbf{v}\|_2 - \text{median}(\|\mathbf{v}\|_2)|$. Subsequently the inlier matches are computed by the constraint:

$$r_i < T\sigma, \tag{12}$$

where T is a scale parameter which is set to 2.5 in our implementation. Subsequently, we estimate the camera pose of j^{th} frame by

$$\min_{\mathbf{C}_j} \sum_i ||\mathbf{x}_{ji} - \pi(\mathbf{C}_j, \mathbf{X}_{j-1,i})|| \tag{13}$$

where $\mathbf{x}_{ji}$ is i^{th} key-point in j^{th} image frame, and $\mathbf{X}_{j-1,i}$ is the corresponding 3D point measured in the previous frame. After setting up the relative pose constraints, we proceed to a pose graph optimization (PGO) using Ceres-solver [24].

VI. Experiments and Results

1) Comparison to Other VO Methods: To evaluate our VO method, we collected five different states of the art methods: 1) ICP, 2) DVO (Stein11) [25], 3) Fovis (Huang11) [7], 4) Whelan et al. (Whel13) [18], and 5) Park et al. (Park17) [26]. Then, we tested their performance on TUM benchmark sequences. Figure 4 and 5 report the performance of the target methods in terms of the relative pose error (RPE) metric. As summarized in Table I, our method outperforms the target methods in terms of mean RPE errors. For example, our method was 5.8% more accurate (or produce less RPE error) than ICP in terms of mean relative translation error. Compared to image based approaches, our method marked 78.8% more accurate than DVO and 59.6% more than FOVIS. Compared to hybrid type VOs, our method achieved approximately 11.5% more than [18] and 34.6% more than [26].

2) Overall Comparison to Other Systems: To evaluate the overall accuracy performance of our system, we quantitatively measured the pose errors w.r.t. the ground truth poses using the absolute trajectory error (ATE) metric. We compared our tracking results with three different state-of-the-art SLAM systems: ORBSLAM2, Elastic-Fusion, and DVOSLAM. In the tests, if the number of tracked camera poses are below 50% of the full frame, we counted the case as a failure.

Figure 6 quantitatively shows the performance of our method with reference to the state-of-the-art works. Table II shows the overall performances of the four tracking systems. Note that our method outperforms Elastic-Fusion and DVOSLAM. ORBSLAM2 is better than our

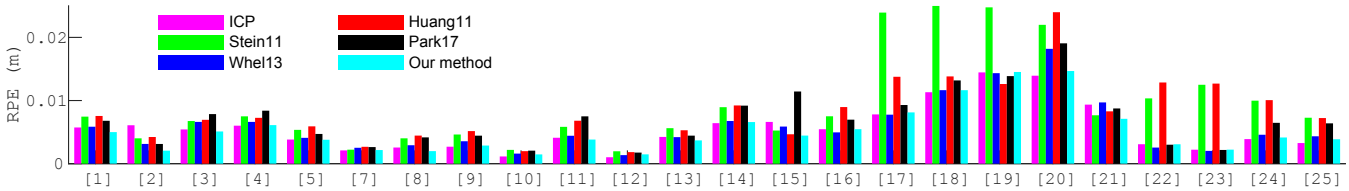


Fig. 4: Relative translation error of VO methods on TUM benchmark: Each data index corresponds to [1]fr1-360, [2]fr1-floor, [3]fr1-desk, [4]fr1-desk2, [5]fr1-room, [6]fr2-360-kidnap, [7]fr2-desk, [8]fr3-long-office, [9]fr1-xyz, [10]fr2-xyz, [11]fr1-rpy, [12]fr2-rpy, [13]fr1-plant, [14]fr1-teddy, [15]fr3-teddy, [16]fr3-cabinet, [17]fr3-large-cabinet, [18]fr3-nostr-notxt-far, [19]fr3-nostrt-notxt-near-loop, [20]fr3-nostrt-txt-far, [21]fr3-nostrt-txt-near-loop, [22]fr3-struct-notxt-far, [23]fr3-struct-notxt-near, [24]fr3-struct-txt-far, and [25]fr3-struct-txt-near', respectively. Note that fr2-360-kidnap is not used for testing the VO algorithms.

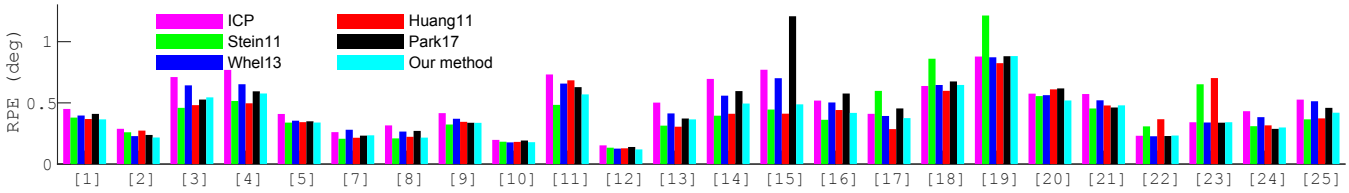


Fig. 5: Relative orientation error of VO methods on TUM benchmark

Method	ICP	Stein 11 [25]	Whelan 13 [18]	Huang 11 [7]	Park 17 [26]	Our method
mean RPE (m)	0.0055	0.0093	0.0058	0.0083	0.0070	**0.0052**
mean RPE (deg)	0.4915	0.4302	0.4494	0.4110	0.4615	**0.4026**

TABLE I: Mean RPE results: *Fr2-360-kidnap* is excluded in this experiment.

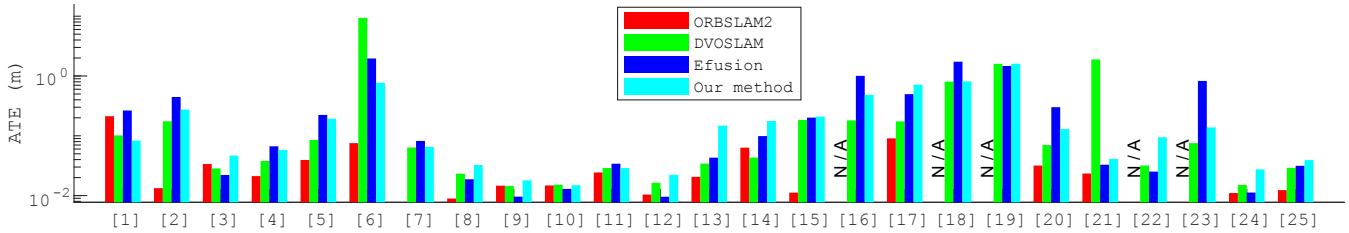


Fig. 6: ATE performances of SLAM systems on TUM benchmark: N/A refers to incomplete tracking. Note that ORBSLAM2 fails on 6 out of the 25 TUM sequences while our method completes them.

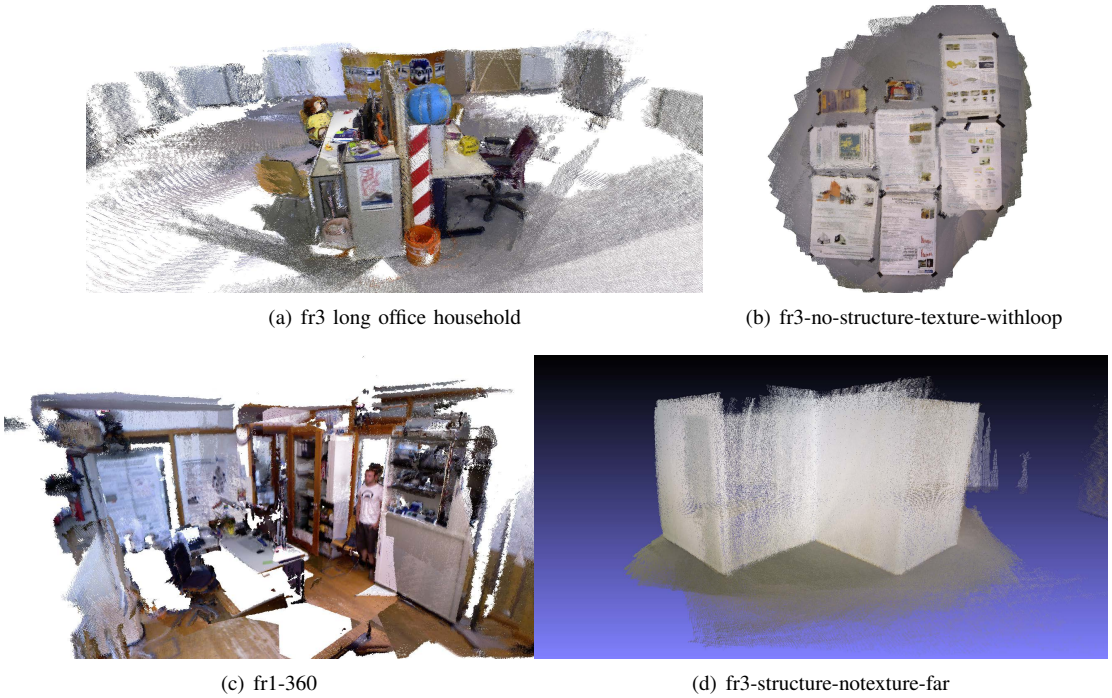
Method	ORBSLAM2 [4]	DVOSLAM [5]	Efusion [27]	Our Method
Mean ATE (Texture-rich Seqs)	**0.0350**	0.1596	0.1266	0.1257
Mean ATE (All 25 Seqs)	N/A	0.6034	0.3773	**0.2516**

TABLE II: Mean ATE for texture-rich sequences and all 25 seqs: ORBSLAM2 was not able to properly handle texture-free sequences while our method was able to complete them. N/A indicates an incomplete tracking.

method on texture-rich sequences but not so much as on texture-free sequences (e.g. *fr3-structure-notexture-far*, *fr3-structure-notexture-near*, *fr3-cabinet*). ORBSLAM2 was not able to initialize a map or failed to complete full tracking in such scenarios. Figure 7 shows the reconstructed 3D point cloud models, which is a byproduct of using our method.

3) Timing Test: Figure 8 reports the timing experiments we have carried out. As it is shown, the speed of our system is highly correlated with BoW matching score β and the window size for dense matching η. In fact, β has much more impact on the tracking results than η because β finds global loop closing candidates while η is only for neighboring frames. Decreasing β causes the system to inspect a larger number of loop closing candidates. Subsequently, the system becomes more robust while loses its efficiency. However, as Figure 8-(c) shows, after a certain point, lowering β does not affect the quality of camera tracking anymore while the computational cost increases. We chose $\beta = 0.1$ and $\eta = 48$ based on these experiments.

We compared our full tracking pipeline with ORBSLAM2



(a) fr3 long office household (b) fr3-no-structure-texture-withloop

(c) fr1-360 (d) fr3-structure-notexture-far

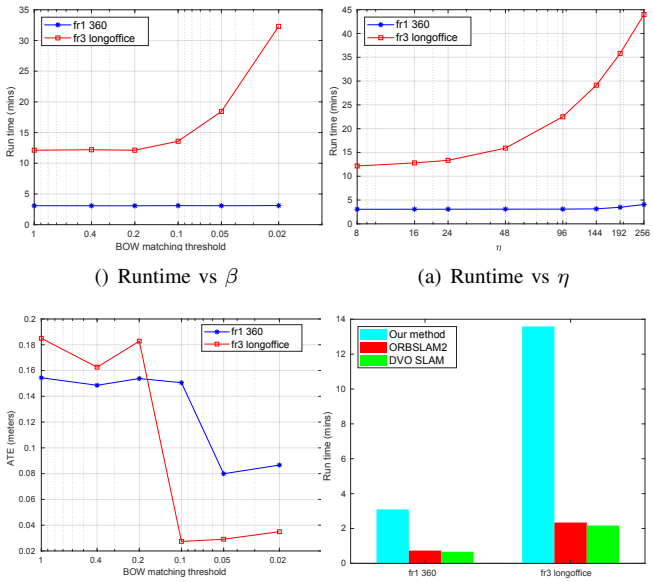
Fig. 7: The final reconstructed 3D models of TUM benchmark sequences

and DVOSLAM on *fr1 360* and *fr3 long office*. Our system is approximately 4.8 times slower than these two methods due to two reasons. First, our VO method uses multiple ICP iterations while our competitive methods do not. This process might be considerably accelerated using GPUs just like Kinect-Fusion [28] and Elastic-Fusion. Also, our method executes semi-dense matching for finding optimum key-frames while other methods use less expensive heuristic or locally optimal decision rules.

VII. Conclusion And Future Works

In this paper, a new RGB-D camera tracking method is presented. As opposed to many existing VSLAM systems, we formulated an algorithmic approach to determine a set of key-frames. Our VO method interpolates DVO and ICP results to estimate a more optimal camera motion while conventional methods solve a joint optimization equation based on a pre-determined weight. The proposed method was validated by performing an extensive experiment using TUM RGB-D hand-held sequences. As it is reported in the paper, our method achieved reliable and robust tracking without employing a full bundle adjustment.

As a current limitation of our system, we observed that constructing an affinity matrix consumes a large amount of computational resources proportional to the growth of SIFT matching queue, which explains relative slower performance compared to existing methods. In the future, we would like to optimize our system using a method like the GPU based cascaded hashing matching [29].



() Runtime vs β (a) Runtime vs η

(b) ATE vs β (c) Runtime comparisons

Fig. 8: Timing experiments: (a) η is fixed to 48 (b) β is fixed to 1.0.

VIII. Acknowledgment

This project was supported by the National Research Foundation(NRF) in South Korea (2017R1A2B3012701 and 2018R1A6A3A11049832).

REFERENCES

[1] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard, "An evaluation of the rgb-d slam system," in *2012 IEEE International Conference on Robotics and Automation*, May 2012, pp. 1691–1696.

[2] C. Kerl, J. Sturm, and D. Cremers, "Robust odometry estimation for rgb-d cameras," in *International Conference on Robotics and Automation (ICRA)*, May 2013.

[3] A. Concha and J. Civera, "RGBDTAM: A cost-effective and accurate RGB-D tracking and mapping system," in *IROS*. IEEE, 2017, pp. 6756–6763.

[4] R. Mur-Artal and J. D. Tardos, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," vol. 33, pp. 1255–1262, 2017.

[5] M. Labbé and F. Michaud, "Rtab-map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation," *Journal of Field Robotics*. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21831

[6] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[7] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Fox, and N. Roy, "Visual odometry and mapping for autonomous flight using an rgb-d camera," in *In Proc. of the Intl. Sym. of Robot. Research*, 2011.

[8] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-d point sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, no. 5, pp. 698–700, Sept 1987.

[9] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 2, pp. 239–256, Feb. 1992. [Online]. Available: http://dx.doi.org/10.1109/34.121791

[10] I. Dryanovski, R. G. Valenti, and J. Xiao, "Fast visual odometry and mapping from rgb-d data," in *2013 IEEE International Conference on Robotics and Automation*, May 2013, pp. 2305–2310.

[11] A. Segal, D. Haehnel, and S. Thrun, "Generalized-icp," in *Proceedings of Robotics: Science and Systems*, Seattle, USA, June 2009.

[12] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A general framework for graph optimization," in *ICRA*, Shanghai, 05/2011 2011.

[13] S. Agarwal, N. Snavely, S. M. Seitz, and R. Szeliski, "Bundle adjustment in the large," in *Proceedings of the 11th European Conference on Computer Vision: Part II*, ser. ECCV'10.

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

[15] O. Vysotska and C. Stachniss, "Effective Visual Place Recognition Using Multi-Sequence Maps," *IEEE Robotics and Automation Letters (RA-L)*, 2019. [Online]. Available: http://www.ipb.uni-bonn.de/pdfs/vysotska2019ral.pdf

[16] Z. Chen, A. Jacobson, N. Sünderhauf, B. Upcroft, L. Liu, C. Shen, I. Reid, and M. Milford, "Deep learning features at scale for visual place recognition," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 3223–3230.

[17] P. Huber, J. Wiley, and W. InterScience, *Robust statistics*. Wiley New York, 1981.

[18] T. Whelan, H. Johannsson, M. Kaess, J. J. Leonard, and J. McDonald, "Robust real-time visual odometry for dense rgb-d mapping," in *2013 IEEE International Conference on Robotics and Automation*, May, pp. 5724–5731.

[19] V. Angladon, S. Gasparini, V. Charvillat, T. Pribanić, T. Petković, M. onlić, B. Ahsan, and F. Bruel, "An evaluation of real-time rgb-d visual odometry algorithms on mobile devices," *Journal of Real-Time Image Processing*, pp. 1–18, 2017.

[20] K. Shoemake, "Animating rotation with quaternion curves," in *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '85.

[21] M. Björkman, N. Bergström, and D. Kragic, "Detecting, segmenting and tracking unknown objects using multi-label mrf inference," *Computer Vision and Image Understanding*, vol. 118, pp. 111–127, 2014.

[22] M. J. Atallah and M. Blanton, Eds., *Algorithms and Theory of Computation Handbook: General Concepts and Techniques*, 2nd ed. Chapman & Hall/CRC, 2010.

[23] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-13, no. 4, pp. 376–380, April 1991.

[24] S. Agarwal, K. Mierle, and Others, "Ceres solver," http://ceres-solver.org.

[25] D. C. Frank Steinbrucker, Jurgen Sturm, "Real-time visual odometry from dense rgb-d images," in *Computer Vision Workshops(ICCV Workshops), IEEE International Conference on*. IEEE, 2011, pp. 1678–1685.

[26] J. Park, Q. Zhou, and V. Koltun, "Colored point cloud registration revisited," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017, pp. 143–152.

[27] T. Whelan, S. Leutenegger, R. S. Moreno, B. Glocker, and A. Davison, "Elasticfusion: Dense slam without a pose graph," in *Proceedings of Robotics: Science and Systems*, Rome, Italy, July 2015.

[28] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon, "Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera," in *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, ser. UIST '11.

[29] J. Cheng, C. Leng, J. Wu, H. Cui, and H. Lu, "Fast and accurate image matching with cascade hashing for 3d reconstruction," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*.