2013 IEEE International Conference on Robotics and Automation (ICRA)
Karlsruhe, Germany, May 6-10, 2013

Point-Plane SLAM for Hand-Held 3D Sensors

Yuichi Taguchi*, Yong-Dian Jian†, Srikumar Ramalingam*, and Chen Feng‡
*Mitsubishi Electric Research Labs (MERL) †Georgia Institute of Technology ‡University of Michigan
*{taguchi, ramalingam} at merl.com †ydjian at gatech.edu ‡cforrest at umich.edu

***Abstract*— We present a simultaneous localization and mapping (SLAM) algorithm for a hand-held 3D sensor that uses both points and planes as primitives. We show that it is possible to register 3D data in two different coordinate systems using any combination of three point/plane primitives (3 planes, 2 planes and 1 point, 1 plane and 2 points, and 3 points). Our algorithm uses the minimal set of primitives in a RANSAC framework to robustly compute correspondences and estimate the sensor pose. As the number of planes is significantly smaller than the number of points in typical 3D data, our RANSAC algorithm prefers primitive combinations involving more planes than points. In contrast to existing approaches that mainly use points for registration, our algorithm has the following advantages: (1) it enables faster correspondence search and registration due to the smaller number of plane primitives; (2) it produces plane-based 3D models that are more compact than point-based ones; and (3) being a global registration algorithm, our approach does not suffer from local minima or any initialization problems. Our experiments demonstrate real-time, interactive 3D reconstruction of indoor spaces using a hand-held Kinect sensor.**

I. Introduction

Interactive 3D reconstruction has always been a useful technique for various applications in robotics, augmented reality, and computer vision. Although there has been several impressive results for real-time sparse [1] and dense [2], [3] 3D reconstruction using a hand-held 2D camera, several challenges such as reconstruction of textureless regions continue to persist. The emergence of inexpensive 3D sensors such as Kinect has addressed this issue. As newer and more exciting applications are being identified to harness the potential of such 3D sensors, two roadblocks are yet to be dismantled:

- **Fast and Accurate 3D Registration:** The limited field of view and resolution of 3D sensors usually result in a partial reconstruction of the entire scene. We need an accurate and fast registration algorithm that fuses successive partial depth maps to model the entire scene.
- **Compact and Semantic Modeling:** The depth maps are usually noisy point clouds, which require a large memory and do not convey any semantic information.

In this paper, we propose a simultaneous localization and mapping (SLAM) algorithm that uses both points and planes as primitives to address these issues. The use of planes along with points enables both faster and more accurate registration than using only points, because the number of planes is much smaller than the number of points in typical 3D data, and planes generated by many points are less affected by measurement noise. Local algorithms such as the iterative-closest point (ICP) algorithm [4] are prone to local minima issues under fast motion of the 3D sensor. Our method performs feature-based global registration and bundle adjustment using both points and planes, and hence does not assume any motion model.

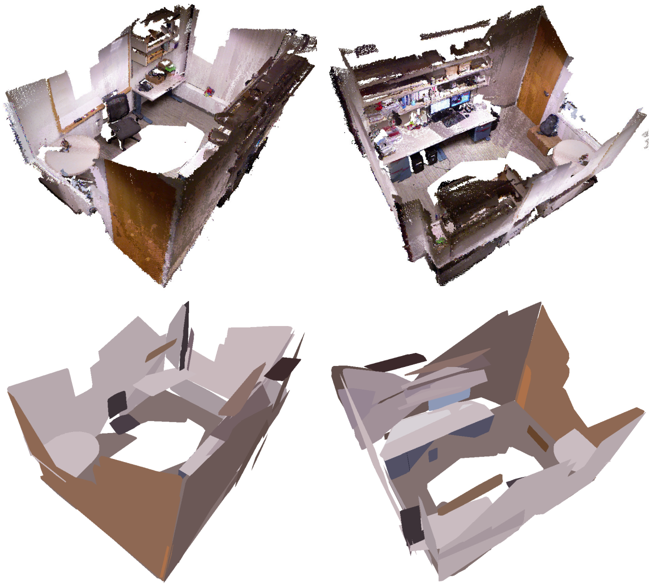

Fig. 1. A 3D model reconstructed from the sequence in Figure 4. Our system not only generates registered 3D point clouds (top), but also reconstructs a scene as a set of planes (bottom). Note that the plane-based model is obtained from plane landmarks, which are generated by our SLAM system in real time, not in post-processing. In this model, the number of keyframes registered is 86, and the numbers of point and plane landmarks are 17290 and 32, respectively.

Note that our system is different from systems such as [5], [6] that extract planes from *registered* 3D point clouds; instead, our system uses planes along with points extracted from individual frames *to register* them. Our approach achieves more efficient registration and generates a plane-based model of the scanned scene, which provides a compact and semantic interpretation of the scene. Figure 1 shows an example of the plane-based model as well as registered point clouds. Note that the plane-based model is generated by our SLAM system in real time, not in a post-processing step.

A. Contributions

We summarize our main contributions:

- We show that any combination of three 3D point/plane primitives allows for registration between two different coordinate systems.

978-1-4673-5643-5/13/$31.00 ©2013 IEEE

- We present a bundle adjustment framework using both 3D points and 3D planes.
- We demonstrate a real-time SLAM system using the proposed techniques with a hand-held Kinect sensor.

B. Related Work

3D-to-3D Registration: Alignment or registration of 3D data is a fundamental problem that has been solved using several techniques. Registration algorithms can be classified into local and global methods. Local methods start with a good initialization and register two 3D point clouds using iterative techniques. The most popular local method is the ICP algorithm [4], which alternates between finding correspondences between 3D points and making local moves using a closed-form solution [7], [8], [9].

Global methods typically consider the entire point cloud, identify some key geometric features, match them across point clouds, and generate an optimal hypothesis using a minimal set of correspondences in a RANSAC framework. The registration obtained by global methods can be refined by local methods. Global methods do not need any initialization, but they suffer from incorrect and insufficient correspondences. The basic geometric primitives used in such global methods are points, lines, and planes. Several registration problems have been studied given both homogeneous and heterogenous correspondences. For example, one can find a closed-form solution for the registration given point-to-point [7], [8], [9], line-to-line [10], plane-to-plane [11], point-to-line [12], point-to-plane [13], and line-to-plane [14] correspondences. Walker et al. [15] presented a closed-form solution using both point-to-point and direction-to-direction (e.g., normal-to-normal) correspondences based on the dual quaternion representation. Olsson et al. [16] described a method obtaining a global optimal solution from point-to-point, point-to-line, and point-to-plane correspondences using branch-and-bound. Li and Hartley [17] used branch-and-bound to obtain the optimal correspondences as well as transformation for the point-to-point registration problem.

SLAM Using 3D Sensors: SLAM using a 2D laser scanner or an array of ultrasonic sensors, which provides 3D data only on a planar slice, has been well studied in mobile robotics for the planar 3 degrees-of-freedom (DOF) motion [18]. More recently, 3D sensors providing full 3D point clouds have been used to compute the 6 DOF motion.

Several variants of the ICP algorithm have been used for frame-by-frame tracking of the 6 DOF sensor motion [19], [20]. KinectFusion [21] extended conventional ICP algorithms by registering a current depth map to a virtual depth map generated from a global truncated signed distance function (TSDF) representation. The TSDF representation integrates all previous depth maps registered into a global coordinate system and enables higher-quality depth map generation than using a single frame, leading to more accurate registration. These ICP-based local registration algorithms work well for slow and smooth motion, but suffer from local minima issues under fast and discontinuous motion.

Henry et al.'s RGB-D mapping system [22] extracts keypoints from RGB images, back-projects them in 3D using the depth maps, and uses 3 point-to-point correspondences to find an initial estimate of the pose using RANSAC, which is further refined using the ICP algorithm. Weingarten and Siegwart [23] presented a SLAM system using plane-to-plane correspondences, which are determined based on motion prediction using odometry or the ICP algorithm. Pathak et al. [24] addressed the plane-to-plane registration problem with unknown correspondences and presented an efficient approach for computing correspondences using geometric constraints between planes. Trevor et al. [25] used a combination of a small field-of-view (FOV) 3D sensor and a large FOV 2D laser scanner for developing a SLAM system using both plane-to-plane and line-to-plane correspondences.

Feature-based global registration methods that solely depend on points [7], [8], [9], [22] suffer from insufficient or incorrect correspondences in textureless regions or regions with repeated patterns. Plane-based methods [11], [23], [24] suffer from degeneracy issues in scenes containing insufficient numbers of non-parallel planes especially if using a limited FOV 3D sensor; Trevor et al. [25] addressed this issue by adding a large FOV laser scanner, but with an additional cost and system complexity. For single-shot 3D sensors like Kinect, line correspondences are hard to obtain because they suffer from noisy or missing depth values especially around depth boundaries.

Driven by these issues, our approach uses both points and planes to avoid the failure modes that are typical while using one of these primitives. This mixed scenario has not been addressed before and we present a closed-form solution to registration using both point-to-point and plane-to-plane correspondences in a unified fashion. To perform global registration without using any motion prediction, we present an efficient RANSAC procedure using geometric constraints between points and planes for solving the correspondence problem.

II. System Overview

Figure 2 shows the flow chart of our SLAM system. The input to the system is a pair of a color image and a depth map (RGB-D data). We use the standard terminology of *measurements* and *landmarks*: Our system extracts measurements from the input data and generates/updates landmarks in a global map. This is done by extracting points and planes from the incoming data and registering them with point/plane landmarks in the global map, generated using the previous measurements. Our main contributions in the system are (1) a RANSAC-based registration algorithm using the minimum number of primitives and (2) a map optimization algorithm using both points and planes, which are detailed in Section III and Section IV, respectively. Here we briefly describe each component of our system.

Point Measurement: Our system extracts 2D keypoints from each color image and back-projects them using the corresponding depth map to obtain 3D point measurements. Each point measurement is represented by $(\boldsymbol{p}_m, \mathcal{D}_m)$, where

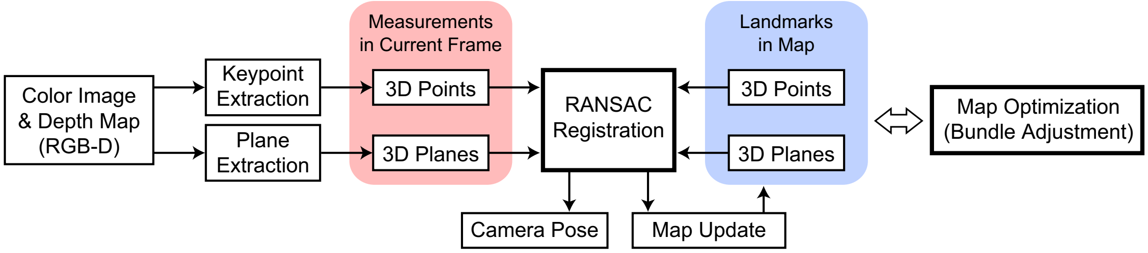

Fig. 2. Overview of our SLAM system. The system extracts 3D points and planes as measurements from each frame, and registers them with the 3D point and plane landmarks in the map to compute the current camera pose. The landmarks in the map are updated using the registered measurements. The landmarks are periodically optimized with our bundle adjustment algorithm using both points and planes.

$\boldsymbol{p}_m$ denotes the 3D position and $\mathcal{D}_m$ denotes the keypoint descriptor. We use SURF implemented in OpenCV as the descriptor.

Plane Measurement: To extract plane primitives from the 3D point cloud generated from the depth map, we use the following multi-stage RANSAC algorithm:

1) Randomly select several (20 in our experiments) reference points in the 3D point cloud.
2) For each reference point, find an optimal plane using nearby points inside a small local window (101 × 101 pixels).
3) Find all inliers (with a threshold of 20 mm) that form a connected component with the reference point with respect to a grid graph on the image space.
4) Identify an optimal plane as the one with maximum and sufficient (>10000) number of inliers.
5) If the optimal plane is found, remove the inliers corresponding to it and return to stage 1). Otherwise, terminate the algorithm.

Each plane measurement is represented by $(\boldsymbol{\pi}_m, \mathcal{I}_m)$, denoting the plane parameters and the inliers, respectively. The 4D vector $\boldsymbol{\pi}_m = (\boldsymbol{n}_m^\mathsf{T}, d_m)^\mathsf{T}$, where $\boldsymbol{n}_m$ is the unit normal vector and d_m is the distance to the origin of the camera coordinate system.

Registration: The pose of the current frame is computed by registering the measurements to the landmarks in the map using our RANSAC-based registration algorithm. Section III presents the closed-form solution for the registration problem using both point-to-point and plane-to-plane correspondences and an efficient RANSAC procedure.

Map Update: In the global map, our system maintains point and plane landmarks generated from measurements in *keyframes*. Our system adds the current frame as a keyframe to the map if its pose computed using the RANSAC-based registration is sufficiently different from poses of previous keyframes. The result of the RANSAC-based registration is also used to determine whether a measurement in the keyframe is associated with an existing landmark (if it is an inlier) or it generates a new landmark (if it is an outlier). Each point landmark is represented by $(\boldsymbol{p}_l, \boldsymbol{\mathcal{D}}_l)$, where $\boldsymbol{p}_l$ denotes the 3D position and $\boldsymbol{\mathcal{D}}_l$ denotes the set of keypoint descriptors obtained from associated point measurements. Each plane landmark is represented by $(\boldsymbol{\pi}_l, \boldsymbol{\mathcal{I}}_l)$, where $\boldsymbol{\pi}_l = (\boldsymbol{n}_l^\mathsf{T}, d_l)^\mathsf{T}$ denotes the plane parameters and $\boldsymbol{\mathcal{I}}_l$ denotes the set of inliers from associated plane measurements.

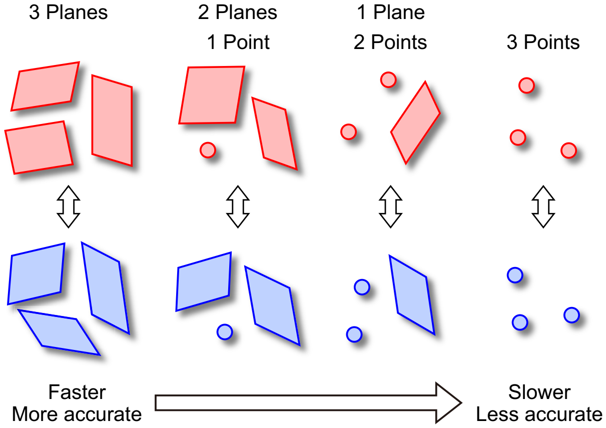

Fig. 3. Registration using the minimal number of point/plane primitives. Our algorithm prefers plane primitives over point primitives, because the number of planes in 3D data is typically smaller than that of points and planes are more robust to noise in 3D data, leading to more accurate registration.

Map Optimization: To have a globally consistent result, our system runs bundle adjustment to optimize the poses of keyframes and the point/plane landmarks using all measurements. This step is explained in Section IV. As in [1], the bundle adjustment runs asynchronously along with the main camera tracking thread.

III. Registration Using Both Points & Planes

In this section, we first present a closed-form registration algorithm using both point-to-point and plane-to-plane correspondences. The algorithm is applicable to 3 or more correspondences; thus it can be used to generate hypotheses in our RANSAC framework, as well as to refine the camera pose with all inliers. We then present our efficient RANSAC procedure, which prioritizes plane primitives over point primitives because the number of planes in 3D data is typically smaller than the number of points. We also describe several geometric constraints to prune false correspondences using an interpretation tree [11].

A. Closed-Form Solution for Mixed Point-to-Point and Plane-to-Plane Correspondences

Let $\{\boldsymbol{p}_i\}$ and $\{\boldsymbol{p}'_i\}$, $i = 1, \dots, M$ be corresponding 3D points, and $\{\boldsymbol{\pi}_j = (\boldsymbol{n}_j{}^\mathsf{T}, d_j)^\mathsf{T}\}$ and $\{\boldsymbol{\pi}'_j = (\boldsymbol{n}'_j{}^\mathsf{T}, d'_j)^\mathsf{T}\}$,

$j = 1, \dots, N$ be corresponding 3D planes in two coordinate systems. We wish to find the rigid body transformation $[\mathsf{R}, \boldsymbol{t}]$ between the coordinate systems.

Solutions for Individual Cases: For the point-to-point correspondence case, it has been shown that the rotation and translation components can be decoupled [7], [8], [9]. Let $\bar{\boldsymbol{p}} = \frac{1}{M}\sum_i \boldsymbol{p}_i$ and $\bar{\boldsymbol{p}}' = \frac{1}{M}\sum_i \boldsymbol{p}'_i$ be the centroids of the 3D point sets, and $\boldsymbol{q}_i = \boldsymbol{p}_i - \bar{\boldsymbol{p}}$ and $\boldsymbol{q}'_i = \boldsymbol{p}'_i - \bar{\boldsymbol{p}}'$. Then the least-squares solution of the rotation is obtained by minimizing

$$\sum_i \|\boldsymbol{q}'_i - \mathsf{R}\boldsymbol{q}_i\|^2. \tag{1}$$

This problem can be solved using the quaternion representation of rotation [7] or singular value decomposition (SVD) [8], [9]. Using the estimated rotation $\hat{\mathsf{R}}$, the translation is computed as the difference between the rotated centroids:

$$\hat{\boldsymbol{t}} = \bar{\boldsymbol{p}}' - \hat{\mathsf{R}}\bar{\boldsymbol{p}}. \tag{2}$$

For the plane-to-plane correspondence case [11], the rotation is obtained by minimizing

$$\sum_j \|\boldsymbol{n}'_j - \mathsf{R}\boldsymbol{n}_j\|^2, \tag{3}$$

which can be solved similar to the case of point-to-point correspondences. For computing the translation, we can stack a linear constraint

$$\boldsymbol{n}'^{\mathsf{T}}_j \boldsymbol{t} = d_j - d'_j \tag{4}$$

for 3 or more planes and solve the linear system.

Solution for Mixed Case: Now we consider the mixed case where we have both point-to-point and plane-to-plane correspondences. We exploit the decoupling used in the individual cases to first compute the rotation and then compute the translation.

To compute the rotation, we combine Eqs. (1) and (3) as

$$\sum_i \|\boldsymbol{q}'_i - \mathsf{R}\boldsymbol{q}_i\|^2 + \sum_j w_j \|\boldsymbol{n}'_j - \mathsf{R}\boldsymbol{n}_j\|^2, \tag{5}$$

where w_j is a weight assigned to the jth plane correspondence. This equation shares the same form as Eqs. (1) and (3), and the optimal rotation is obtained in the same manner. Specifically, we define a 3×3 correlation matrix K [26] as

$$\mathsf{K} = \sum_i \boldsymbol{q}'_i \boldsymbol{q}_i^{\mathsf{T}} + \sum_j w_j \boldsymbol{n}'_j \boldsymbol{n}_j^{\mathsf{T}}. \tag{6}$$

Let $\mathsf{K} = \mathsf{UDV}^{\mathsf{T}}$ be the singular value decomposition of K, where $\mathsf{D} = \mathrm{diag}(\sigma_1, \sigma_2, \sigma_3)$, $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq 0$. Then the optimal rotation $\hat{\mathsf{R}}$ is given by [9], [26]

$$\hat{\mathsf{R}} = \mathsf{U} \begin{pmatrix} 1 & & \\ & 1 & \\ & & \det(\mathsf{UV}^{\mathsf{T}}) \end{pmatrix} \mathsf{V}^{\mathsf{T}}. \tag{7}$$

To compute the translation, we minimize

$$M\|\boldsymbol{t} - (\bar{\boldsymbol{p}}' - \hat{\mathsf{R}}\bar{\boldsymbol{p}})\|^2 + \sum_j w_j \left(\boldsymbol{n}'^{\mathsf{T}}_j \boldsymbol{t} - (d_j - d'_j)\right)^2. \tag{8}$$

This corresponds to defining a linear system

$$\underbrace{\begin{pmatrix} \mathsf{A}_1 \\ \mathsf{A}_2 \end{pmatrix}}_{\mathsf{A}} \boldsymbol{t} = \underbrace{\begin{pmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \end{pmatrix}}_{\boldsymbol{b}}, \tag{9}$$

$$\begin{aligned} \mathsf{A}_1 &= M\mathsf{I}_3\ , & \boldsymbol{b}_1 &= M(\bar{\boldsymbol{p}}' - \hat{\mathsf{R}}\bar{\boldsymbol{p}})\ , \\ \mathsf{A}_2 &= \begin{pmatrix} w_1 \boldsymbol{n}'^{\mathsf{T}}_1 \\ \vdots \\ w_N \boldsymbol{n}'^{\mathsf{T}}_N \end{pmatrix}, & \boldsymbol{b}_2 &= \begin{pmatrix} w_1(d_1 - d'_1) \\ \vdots \\ w_N(d_N - d'_N) \end{pmatrix}, \end{aligned} \tag{10}$$

where I_3 is the 3×3 identity matrix. A_1 and $\boldsymbol{b}_1$ disappear if the number of point correspondences M is zero, while A_2 and $\boldsymbol{b}_2$ disappear if the number of plane correspondences N is zero. The least-squares solution is given by

$$\hat{\boldsymbol{t}} = (\mathsf{A}^{\mathsf{T}}\mathsf{A})^{-1}\mathsf{A}^{\mathsf{T}}\boldsymbol{b}. \tag{11}$$

B. Degeneracy Issues

To uniquely extract R and $\boldsymbol{t}$, the correlation matrix K in Eq. (6) and the matrix A in Eq. (9) should satisfy certain conditions. To uniquely compute R, the rank of K should be greater than 1 and at least one of the following conditions must hold true [26]: (1) $\det(\mathsf{UV}^{\mathsf{T}}) = 1$; (2) The minimum singular value of K is a simple root ($\sigma_2 \neq \sigma_3$). For the translation $\boldsymbol{t}$ to be uniquely determined, the matrix A should be rank 3. Note that if there is at least one point correspondence ($M > 0$), the rank of A is 3 (because of A_1).

The matrices K and A satisfy the above properties if the correspondences possess at least one of the following as shown in Figure 3:

1) 3 non-colinear points ($\mathrm{rank}(\mathsf{K}) = 2$).
2) 2 points and 1 plane where the vector joining the two points is not parallel to the normal of the plane ($\mathrm{rank}(\mathsf{K}) = 2$).
3) 1 point and 2 non-parallel planes ($\mathrm{rank}(\mathsf{K}) = 2$).
4) 3 planes whose normals span $\mathbb{R}^3$ ($\mathrm{rank}(\mathsf{A}_2) = 3$, $\mathrm{rank}(\mathsf{K}) = 3$).

C. Efficient RANSAC Procedure

In contrast to correspondences in the 2D image space, the 3D primitives provide several invariants that can be used to prune false correspondences. These invariants are the geometric entities that are independent of the coordinate systems. We identify the following three invariants:

- **I1**: Distance between two points.
- **I2**: Distance between a point and a plane.
- **I3**: Angle between two plane normals.

Invariants can be denoted by a vector $\boldsymbol{I} = (i_1, i_2, i_3)$, where i_1, i_2, and i_3 represent the number of invariants of the types **I1**, **I2**, and **I3**, respectively. We can observe that all the corresponding triplets involving points and planes possess a total of three invariants as shown below:

- 3 points: $\boldsymbol{I} = (3, 0, 0)$
- 1 plane and 2 points: $\boldsymbol{I} = (1, 2, 0)$
- 2 planes and 1 point: $\boldsymbol{I} = (0, 2, 1)$
- 3 planes: $\boldsymbol{I} = (0, 0, 3)$

In our system, we use an interpretation tree [11] to prune false correspondences using these invariants.

Prior to the pruning based on invariants, we need to obtain some initial candidates of correspondences. In the case of points, we obtain such candidates by using nearest-neighbor descriptor matching between point measurements in the current frame and point landmarks in the map. In the case of planes, we use all possible combinations as candidates. We start the RANSAC procedure using 3-plane correspondences, because the number of planes in 3D data is usually much smaller than that of keypoints. Moreover, since planes are generated by many points, they are less affected by the noise in 3D data, leading to more accurate registration. If the measurements include less than 3 planes or if the number of inliers computed using 3-plane correspondences is small, then we try 2-plane and 1-point correspondences and so on, as shown in Figure 3. We terminate the RANSAC procedure by using a standard criteria on the minimum number of triplets required to be sampled [27].

IV. Bundle Adjustment Using Both Points & Planes

Conventional bundle adjustment approaches simultaneously optimize the poses of all keyframes and/or point landmarks using measurements obtained from 2D images [1], [28] or 3D sensors [29], [30], [31]. We extend such approaches for 3D sensors by adding plane landmarks: We simultaneously optimize point and plane landmarks as well as the poses of all keyframes.

We denote the variables to be optimized as

- Point landmarks: $\boldsymbol{p}_l^i = (x^i, y^i, z^i)^\mathsf{T}$
- Plane landmarks: $\boldsymbol{\pi}_l^j = (a^j, b^j, c^j, d^j)^\mathsf{T}$
- Keyframe poses: $\boldsymbol{T}^k = (t_x^k, t_y^k, t_z^k, \theta_x^k, \theta_y^k, \theta_z^k)^\mathsf{T}$

Here, $\boldsymbol{t}^k = (t_x^k, t_y^k, t_z^k)^\mathsf{T}$ are the (x, y, z) components of the translation of the kth keyframe, and $\boldsymbol{\theta}^k = (\theta_x^k, \theta_y^k, \theta_z^k)^\mathsf{T}$ represent the rotation around the (x, y, z) axes. The rotation matrix R^k of the kth keyframe is represented by $\mathsf{R}^k = \mathsf{R}_z(\theta_z^k)\mathsf{R}_y(\theta_y^k)\mathsf{R}_x(\theta_x^k)$. We compute the Jacobian matrices using the point and plane measurements as follows.

Point Landmarks: For point landmarks, we minimize the distance error between a point landmark $\boldsymbol{p}_l^i$ and an associated point measurement $\boldsymbol{p}_m^k = (x_m^k, y_m^k, z_m^k)^\mathsf{T}$ in the kth keyframe, which is expressed as

$$\left\| \boldsymbol{p}_l^i - \left(\mathsf{R}^k \boldsymbol{p}_m^k + \boldsymbol{t}^k\right) \right\|^2 = 0. \tag{12}$$

Using the current estimate of the landmark $\hat{\boldsymbol{p}}_l^i = (\hat{x}^i, \hat{y}^i, \hat{z}^i)^\mathsf{T}$ and the keyframe pose $[\hat{\mathsf{R}}^k, \hat{\boldsymbol{t}}^k]$, we linearize the equation as

$$\left\| \hat{\boldsymbol{p}}_l^i + \Delta \boldsymbol{p}_l^i - \left(\Delta\mathsf{R}^k \hat{\boldsymbol{p}}_m^k + \Delta \boldsymbol{t}^k\right) \right\|^2 = 0, \tag{13}$$

where $\hat{\boldsymbol{p}}_m^k = \hat{\mathsf{R}}^k \boldsymbol{p}_m^k + \hat{\boldsymbol{t}}^k$ and

$$\Delta\mathsf{R}^k = \begin{pmatrix} 1 & -\Delta\theta_z^k & \Delta\theta_y^k \\ \Delta\theta_z^k & 1 & -\Delta\theta_x^k \\ -\Delta\theta_y^k & \Delta\theta_x^k & 1 \end{pmatrix}. \tag{14}$$

From Eq. (13), we obtain 3 equations separately for each (x, y, z) component. The equation for the x component is

$$\begin{pmatrix} 2(\hat{x}^i - \hat{x}_m^k) \\ 0 \\ 0 \\ 2(\hat{x}_m^k - \hat{x}^i) \\ 0 \\ 0 \\ 0 \\ 2\hat{z}_m^k(\hat{x}_m^k - \hat{x}^i) \\ 2\hat{y}_m^k(\hat{x}^i - \hat{x}_m^k) \end{pmatrix}^\mathsf{T} \begin{pmatrix} \Delta x^i \\ \Delta y^i \\ \Delta z^i \\ \Delta t_x^k \\ \Delta t_y^k \\ \Delta t_z^k \\ \Delta\theta_x^k \\ \Delta\theta_y^k \\ \Delta\theta_z^k \end{pmatrix} = -(\hat{x}^i - \hat{x}_m^k)^2, \tag{15}$$

and those for the y and z components can be similarly obtained.

Plane Landmarks: For plane landmarks, instead of minimizing an algebraic error between planes, we minimize a geometric error defined by the sum of distances between a plane landmark and 3D points sampled from associated plane measurements in a keyframe. Specifically, we uniformly sample 3D points $\boldsymbol{x}_m^{k,s}$ from inlier 3D points of a plane measurement $\boldsymbol{\pi}_m^k$, and compute the distance between each sampled point and the associated plane landmark $\boldsymbol{\pi}_l^j$. The minimization is expressed as

$$\sum_s \left(\boldsymbol{\pi}_l^j\right)^\mathsf{T} \begin{pmatrix} \mathsf{R}^k \boldsymbol{x}_m^{k,s} + \boldsymbol{t}^k \\ 1 \end{pmatrix} = 0. \tag{16}$$

We linearize the equation using the current estimate of the plane landmark $\hat{\boldsymbol{\pi}}_l^j = (\hat{a}^j, \hat{b}^j, \hat{c}^j, \hat{d}^j)^\mathsf{T}$ and the keyframe pose $[\hat{\mathsf{R}}^k, \hat{\boldsymbol{t}}^k]$ as

$$\sum_s \left(\hat{\boldsymbol{\pi}}_l^j + \Delta\boldsymbol{\pi}_l^j\right)^\mathsf{T} \begin{pmatrix} \Delta\mathsf{R}^k \hat{\boldsymbol{x}}_m^{k,s} + \Delta\boldsymbol{t}^k \\ 1 \end{pmatrix} = 0, \tag{17}$$

where $\hat{\boldsymbol{x}}_m^{k,s} = \hat{\mathsf{R}}^k \boldsymbol{x}_m^{k,s} + \hat{\boldsymbol{t}}^k = (\hat{x}_m^{k,s}, \hat{y}_m^{k,s}, \hat{z}_m^{k,s})^\mathsf{T}$. After simplifying the equation, we have

$$\sum_s \begin{pmatrix} \hat{x}_m^{k,s} \\ \hat{y}_m^{k,s} \\ \hat{z}_m^{k,s} \\ 1 \\ \hat{a}^j \\ \hat{b}^j \\ \hat{c}^j \\ \hat{c}^j \hat{y}_m^{k,s} - \hat{b}^j \hat{z}_m^{k,s} \\ \hat{a}^j \hat{z}_m^{k,s} - \hat{c}^j \hat{x}_m^{k,s} \\ \hat{b}^j \hat{x}_m^{k,s} - \hat{a}^j \hat{y}_m^{k,s} \end{pmatrix}^\mathsf{T} \begin{pmatrix} \Delta a^j \\ \Delta b^j \\ \Delta c^j \\ \Delta d^j \\ \Delta t_x^k \\ \Delta t_y^k \\ \Delta t_z^k \\ \Delta\theta_x^k \\ \Delta\theta_y^k \\ \Delta\theta_z^k \end{pmatrix} = -\sum_s \left(\hat{\boldsymbol{\pi}}_l^j\right)^\mathsf{T} \begin{pmatrix} \hat{\boldsymbol{x}}_m^{k,s} \\ 1 \end{pmatrix}. \tag{18}$$

Solution: By stacking Eqs. (15) and (18) for all point/plane landmarks and keyframes, we obtain a linear system $\mathsf{J}\boldsymbol{\Delta} = -\boldsymbol{\epsilon}_0$, where J represents the Jacobian matrix, $\boldsymbol{\epsilon}_0$ is the error vector given the current estimates of the variables, and $\boldsymbol{\Delta}$ is the update vector consisting of $\Delta\boldsymbol{p}_l^i$, $\Delta\boldsymbol{\pi}_l^j$, and $\Delta\boldsymbol{T}^k$ to be computed. Our system currently computes the entire linear system whenever the bundle adjustment process is called, and solves it using the Gauss-Newton iteration method with a sparse linear solver in Eigen[1]. It

[1] `http://eigen.tuxfamily.org`

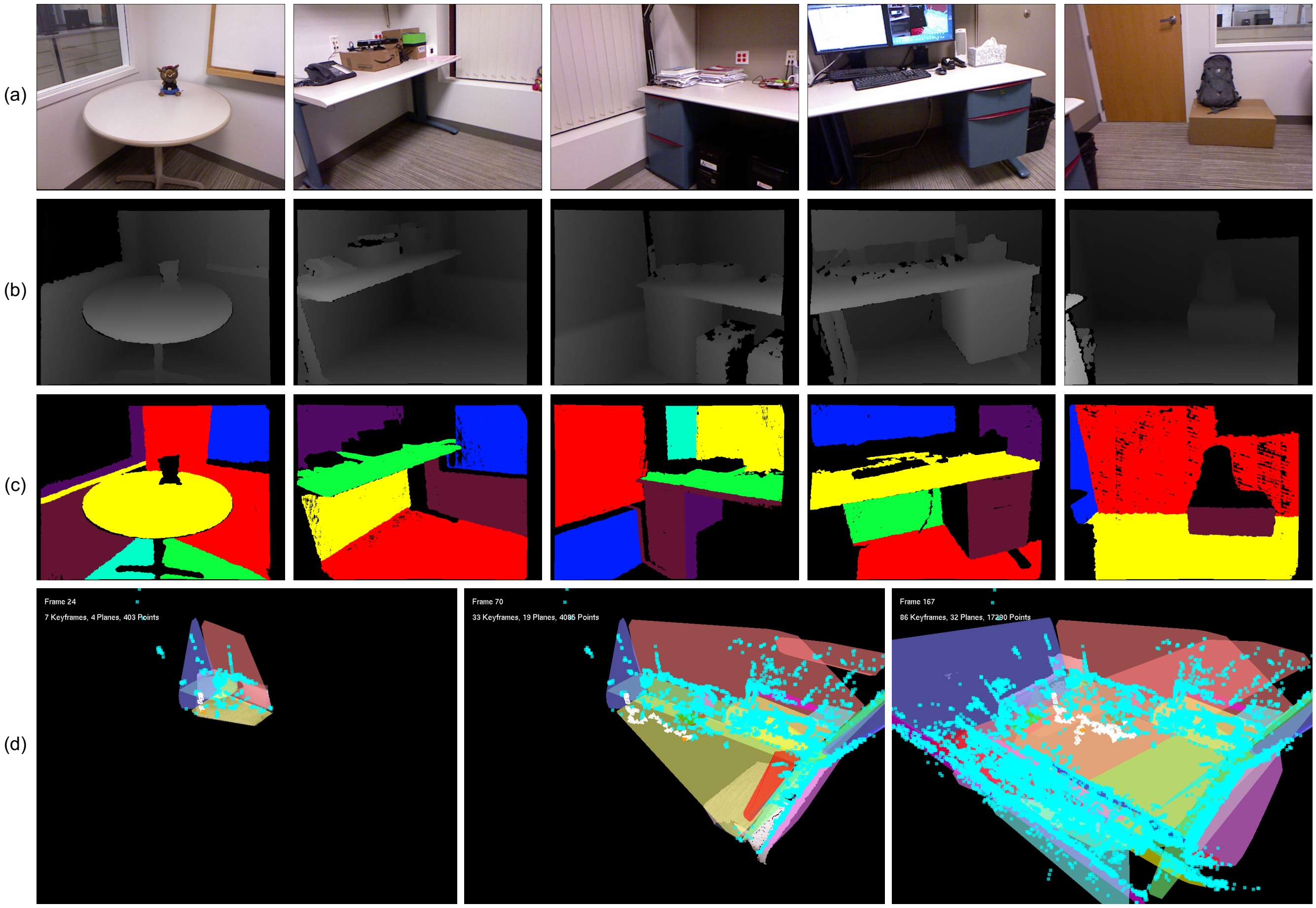
(a)
(b)
(c)
(d)
Frame 24
7 Keyframes, 4 Planes, 403 Points
Frame 70
33 Keyframes, 19 Planes, 4085 Points
Frame 167
86 Keyframes, 32 Planes, 17290 Points

Fig. 4. An example of real-time 3D reconstruction using a hand-held Kinect. (a) Color images and (b) depth maps from the captured sequence. (c) Our system performs plane fitting and segmentation to extract plane measurements from the depth maps. Each plane measurement is depicted with different colors. (d) Snapshots of our interactive visualization system. Plane landmarks (transparent polygons with different colors) and point landmarks (cyan points) are superimposed on the current frame (colored point cloud), demonstrating the correct registration. White camera icons represent the poses of keyframes, while the orange one shows the current pose. Please refer to the supplementary video demonstrating results for the entire sequence.

could be made more efficient by using hierarchical [30] or tree-based [31] graph structures that allow us to compute incremental updates and solutions of the linear system.

V. Experiments and Discussion

We use a Kinect sensor that provides color images and depth maps at a resolution of 640 × 480 pixels. Figure 4 shows a sequence captured in an office room. Figures 4(a) and 4(b) show examples of the color images and depth maps from the sequence. Figure 4(c) depicts plane extraction results from the corresponding depth maps. Figure 4(d) shows snapshots of our interactive visualization of the SLAM result. Results for the entire sequence are available in the supplementary video.

Our system superimposes the point and plane landmarks onto the current point cloud when the registration is successful. If the registration fails, our system indicates it to the user (see the supplementary video). Note that our system always performs global registration of the current frame with respect to the map. Thus registration failure of any frame does not affect subsequent ones; the user can simply bring the sensor to a location from which a part of the reconstructed map is observable.

A 3D model reconstructed from the sequence is shown in Figure 1. Figure 5 shows other reconstruction examples. In addition to registered point clouds, our system provides reconstructed plane landmarks as a plane-based model of the scene, which is more compact and provides semantic information.

Note that in Figure 4(d) and in the video, several planes are visualized as a single colored plane even if they are not physically connected (e.g., two table tops at the same height). This is because our registration algorithm handles planes as infinite planes; the algorithm associates such planes to a single plane landmark for better registration. Nevertheless, we maintain plane inliers for each plane measurement and use them to compute plane boundaries in the reconstructed models; thus those planes are separately depicted in the models shown in Figures 1 and 5.

Comparison: Figure 6 compares the 3D models reconstructed by our approach and the conventional approach that uses only point correspondences. After scanning the entire room and returning to the initial location (the round table),

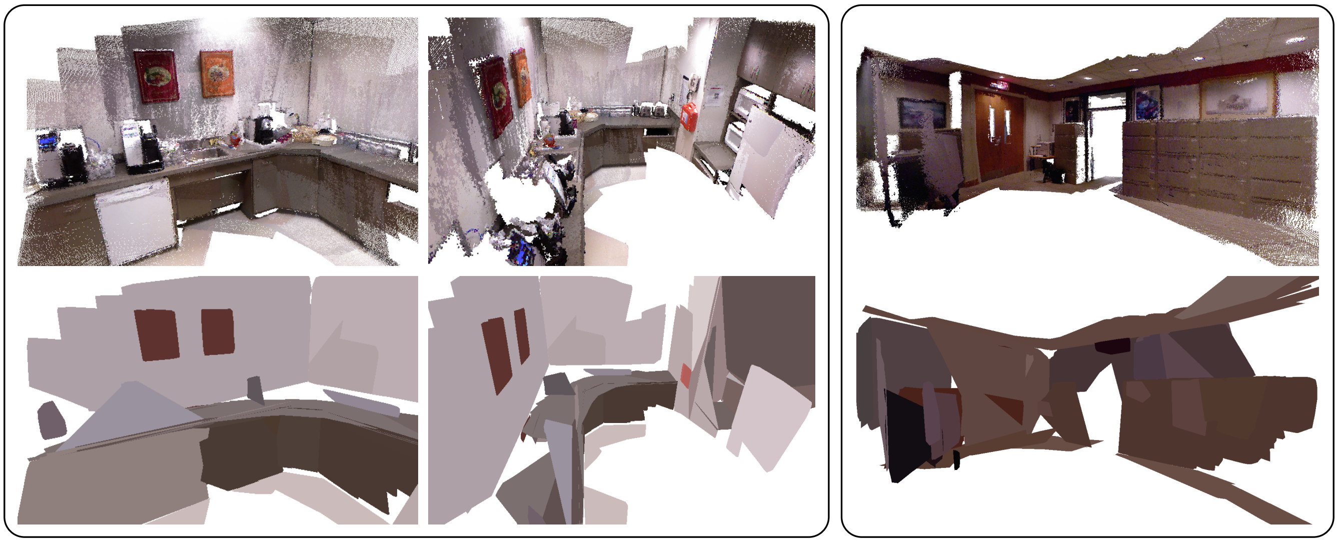

Fig. 5. Reconstruction results for different scenes, depicted as (top) registered point clouds and (bottom) plane-based models. The result on the left was generated by registering 44 keyframes, including 8986 point and 39 plane landmarks. The result on the right was generated by registering 120 keyframes, including 7301 point and 30 plane landmarks.

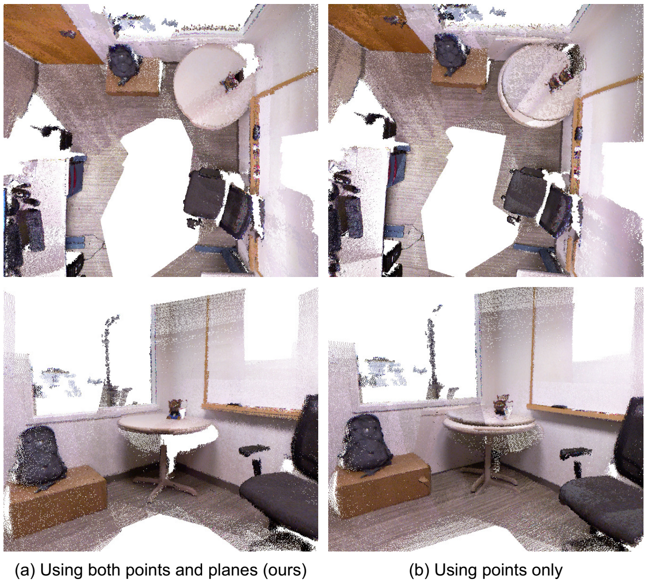

Fig. 6. Comparison of SLAM results obtained using (a) both points and planes (ours) and (b) only points. Note that we started the SLAM process at the location of the round table, scanned the entire room, and returned to the start location as shown in the supplementary video. The result obtained using only points exhibits drifts, appearing as the ghosting artifacts of the table, while our result shows correct registration.

the conventional approach using only points produced drifts, appearing as the ghosting artifacts of the round table in Figure 6, while our approach maintained correct registration. This is because (1) plane correspondences are more accurate and stable than point correspondences especially in textureless regions and regions with repeated patterns; and (2) plane correspondences can provide long-range interactions among several frames (e.g., frames including the floor are associated with a single plane landmark of the floor), leading to globally consistent registration. Note that the conventional point-based approach could correct the misalignment with a refinement step using the ICP algorithm, but with additional computational costs. Moreover, in some cases where our approach found a good solution, the conventional point-based approach failed without providing any hypothesis.

TABLE I

PROCESSING TIME FOR EACH COMPONENT AVERAGED OVER THE SEQUENCE SHOWN IN FIGURE 4 (IN MSEC).

Point Measurement Extraction	129
Plane Measurement Extraction	210
RANSAC Registration	138
Other (Map Update, Data Copy)	5
Total	482

TABLE II

NUMBER AND PERCENTAGE OF EACH REGISTRATION TYPE SELECTED OVER THE SEQUENCE SHOWN IN FIGURE 4.

3 Planes	2 Planes 1 Point	1 Plane 2 Points	3 Points	Total Registration
31 (22%)	60 (42%)	48 (33%)	5 (3%)	144

Processing Time and Analysis: Currently our system runs at 2–3 frames per second, depending on the number of landmarks, on a standard PC with Intel Core i7-950 processor. Table I summarizes the average processing time for each component of the system over the sequence shown in Figure 4. Map optimization using bundle adjustment took up to 10 seconds depending on the accuracy of the initial solution and the number of variables, increasing as the system adds more keyframes. However, it does not affect the system frame rate due to the asynchronous update. Currently the majority of the processing time is spent by the point/plane measurement extraction. By using faster keypoint and plane extraction algorithms [32], [33] we could further improve the speed of our system.

Table II shows the number of each registration type selected out of 144 successful RANSAC registration trials, demonstrating that our system prefers plane primitives over point primitives. Nevertheless, using only planes (i.e., 3 planes) is not enough for many cases, because there are frames containing only a few non-degenerate planes due to the limited field of view of the 3D sensor.

VI. Conclusions

We presented a real-time SLAM system for hand-held 3D sensors that uses both point and plane primitives for registration. This mixed approach enables faster and more accurate registration than using only points. Our system generates a 3D model as a set of planes, which provides more compact and semantic information of the scene than point-based representations.

Acknowledgments: We thank Jay Thornton, Amit Agrawal, Tim K. Marks, and Esra Ataer-Cansizoglu for their helpful comments and feedback. This work was supported by and done at MERL. Yong-Dian Jian and Chen Feng contributed to the work while they were interns at MERL. A preliminary version of this paper appeared in [34].

References

[1] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. IEEE Int'l Symp. Mixed and Augmented Reality (ISMAR)*, Nov. 2007, pp. 1–10.

[2] R. A. Newcombe and A. J. Davison, "Live dense reconstruction with a single moving camera," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, June 2010, pp. 1498–1505.

[3] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "DTAM: Dense tracking and mapping in real-time," in *Proc. IEEE Int'l Conf. Computer Vision (ICCV)*, Nov. 2011, pp. 2320–2327.

[4] 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.

[5] R. Schnabel, R. Wahl, and R. Klein, "Efficient RANSAC for point-cloud shape detection," *Computer Graphics Forum*, vol. 26, no. 2, pp. 214–226, June 2007.

[6] M. F. Fallon, H. Johannsson, and J. J. Leonard, "Efficient scene simulation for robust Monte Carlo localization using an RGB-D camera," in *Proc. IEEE Int'l Conf. Robotics Automation (ICRA)*, May 2012, pp. 1663–1670.

[7] B. K. P. Horn, "Closed-form solution of absolute orientation using unit quaternions," *J. Opt. Soc. Am. A*, vol. 4, no. 4, pp. 629–642, Apr. 1987.

[8] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-D point sets," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 9, no. 5, pp. 698–700, May 1987.

[9] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, no. 4, pp. 376–380, Apr. 1991.

[10] Z. Zhang and O. D. Faugeras, "Determining motion from 3D line segment matches: A comparative study," *Image and Vision Computing*, vol. 9, no. 1, pp. 10–19, Feb. 1991.

[11] W. E. L. Grimson and T. Lozano-Pérez, "Model-based recognition and localization from sparse range or tactile data," MIT AI Lab, A. I. Memo 738, Aug. 1983.

[12] D. Nistér, "A minimal solution to the generalised 3-point pose problem," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, vol. 1, June 2004, pp. 560–567.

[13] S. Ramalingam and Y. Taguchi, "A theory of minimal 3D point to 3D plane registration and its generalization," *Int'l J. Computer Vision*, Sept. 2012.

[14] H. H. Chen, "Pose determination from line-to-plane correspondences: Existence condition and closed-form solutions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, no. 6, pp. 530–541, June 1991.

[15] M. W. Walker, L. Shao, and R. A. Volz, "Estimating 3-D location parameters using dual number quaternions," *CVGIP: Image Understanding*, vol. 54, no. 3, pp. 358–367, Nov. 1991.

[16] C. Olsson, F. Kahl, and M. Oskarsson, "The registration problem revisited: Optimal solutions from points, lines and planes," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, June 2006, pp. 1206–1213.

[17] H. Li and R. Hartley, "The 3D-3D registration problem revisited," in *Proc. IEEE Int'l Conf. Computer Vision (ICCV)*, Oct. 2007.

[18] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. The MIT Press, 2005.

[19] S. Rusinkiewicz, O. Hall-Holt, and M. Levoy, "Real-time 3D model acquisition," *ACM Trans. Graphics*, vol. 21, no. 3, pp. 438–446, July 2002.

[20] F. Pomerleau, S. Magnenat, F. Colas, M. Liu, and R. Siegwart, "Tracking a depth camera: Parameter exploration for fast ICP," in *Proc. IEEE/RSJ Int'l Conf. Intelligent Robots and Systems (IROS)*, Sept. 2011, pp. 3824–3829.

[21] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon, "KinectFusion: Real-time dense surface mapping and tracking," in *Proc. IEEE Int'l Symp. Mixed and Augmented Reality (ISMAR)*, Oct. 2011, pp. 127–136.

[22] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments," in *Proc. Int'l Symp. Experimental Robotics (ISER)*, 2010.

[23] J. Weingarten and R. Siegwart, "3D SLAM using planar segments," in *Proc. IEEE/RSJ Int'l Conf. Intelligent Robots and Systems (IROS)*, Oct. 2006, pp. 3062–3067.

[24] K. Pathak, A. Birk, N. Vaškevičius, and J. Poppinga, "Fast registration based on noisy planes with unknown correspondences for 3-D mapping," *IEEE Trans. Robotics*, vol. 26, no. 3, pp. 424–441, June 2010.

[25] A. J. B. Trevor, J. G. Rogers III, and H. I. Christensen, "Planar surface SLAM with 3D and 2D sensors," in *Proc. IEEE Int'l Conf. Robotics Automation (ICRA)*, May 2012, pp. 3041–3048.

[26] K. Kanatani, "Unbiased estimation and statistical analysis of 3-D rigid motion from two views," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 15, no. 1, pp. 37–50, Jan. 1993.

[27] R. Raguram, J.-M. Frahm, and M. Pollefeys, "A comparative analysis of RANSAC techniques leading to adaptive real-time random sample consensus," in *Proc. European Conf. Computer Vision (ECCV)*, vol. 2, Oct. 2008, pp. 500–513.

[28] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment — a modern synthesis," in *Proc. Int'l Workshop on Vision Algorithms: Theory and Practice*, 2000, pp. 298–372.

[29] F. Lu and E. Milios, "Globally consistent range scan alignment for environment mapping," *Autonomous Robots*, vol. 4, no. 4, pp. 333–349, Oct. 1997.

[30] G. Grisetti, R. Kümmerle, C. Stachniss, U. Frese, and C. Hertzberg, "Hierarchical optimization on manifolds for online 2D and 3D mapping," in *Proc. IEEE Int'l Conf. Robotics Automation (ICRA)*, May 2010, pp. 273–278.

[31] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping using the bayes tree," *Int'l J. Robotics Research*, vol. 31, no. 2, pp. 216–235, Feb. 2012.

[32] D. Holz, S. Holzer, R. B. Rusu, and S. Behnke, "Real-time plane segmentation using RGB-D cameras," in *Proc. RoboCup Symposium*, July 2011.

[33] K. Georgiev, R. T. Creed, and R. Lakaemper, "Fast plane extraction in 3D range data based on line segments," in *Proc. IEEE/RSJ Int'l Conf. Intelligent Robots and Systems (IROS)*, Sept. 2011, pp. 3808–3815.

[34] Y. Taguchi, Y.-D. Jian, S. Ramalingam, and C. Feng, "SLAM using both points and planes for hand-held 3D sensors," in *Proc. IEEE Int'l Symp. Mixed and Augmented Reality (ISMAR), Poster*, Nov. 2012, pp. 321–322.