Structured Global Registration of RGB-D Scans in Indoor Environments

Maciej Halber
Princeton University

Thomas Funkhouser
Princeton University

Abstract

RGB-D scanning of indoor environments (offices, homes, museums, etc.) is important for a variety of applications, including on-line real estate, virtual tourism, and virtual reality. To support these applications, we must register the RGB-D images acquired with an untracked, hand-held camera into a globally consistent and accurate 3D model. Current methods work effectively for small environments with trackable features, but often fail to reproduce large-scale structures (e.g., straight walls along corridors) or long-range relationships (e.g., parallel opposing walls in an office). In this paper, we investigate the idea of integrating a structural model into the global registration process. We introduce a fine-to-coarse algorithm that detects planar structures spanning multiple RGB-D frames and establishes geometric constraints between them as they become aligned. Detection and enforcement of these structural constraints in the inner loop of a global registration algorithm guides the solution towards more accurate global registrations, even without detecting loop closures. During experiments with a newly created benchmark for the SUN3D dataset, we find that this approach produces registration results with greater accuracy and better robustness than previous alternatives.

1 Introduction

With the proliferation of inexpensive RGB-D video cameras, there are now great opportunities for systems to capture 3D geometric models of real-world indoor environments for later visualization, analysis, storage, and editing [CLH15]. Potential applications include on-line real estate, virtual tourism, home remodeling, training, simulation, and virtual reality.

Motivated by these applications, our goal is to build a system that takes a sequence of RGB-D images captured with a hand-held video camera as input, and produces a geometrically detailed, globally consistent, geospatially accurate, and textured 3D model as output. We would like the system to work robustly in a wide range of static indoor environments (offices, homes, museums, etc.), execute off-line within practical computational limits (runs on a laptop within minutes or hours), and work automatically with inexpensive commodity cameras so that it can be used by non-experts.

It is difficult to register RGB-D data acquired with an untracked, hand-held camera into a globally consistent and accurate 3D model. Though RGB-D camera poses can usually be tracked precisely over short distances [NDI^{+}11], tracking often fails in texture-less regions and/or drifts over long ranges [NZIS13]. Loop closure and global optimization can mitigate these issues when distinctive features are observed multiple times in a scan [HKH^{+}10]. However, state-of-the-art global registration systems still produce warped surfaces and improbable global structures (e.g., walls that are not parallel/perpendicular to one another, as shown in Figure 8). Though it might be possible to hide these errors during a later surface reconstruction [FG14] or model fitting phase [MMBM15], post-processing the data as a point cloud does not address the underlying problem: inaccurate global registration.

We address this problem by integrating detection and enforcement of a structural model into the inner loop of a global registration algorithm. Specifically, we detect planar structures, infer structural relationships between them (coplanarity, parallelism, perpendicularity), and add soft constraints enforcing those relationships into each iteration of a global ICP algorithm. The advantage of this approach is that the detected high-level geometric structure can directly influence the camera pose estimations, leading to more accurate and robust results even

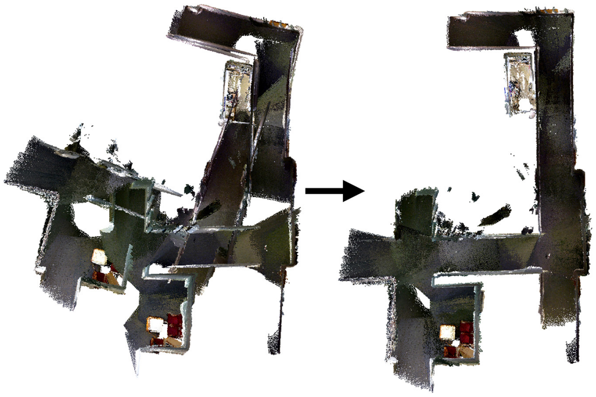


Figure 1: Our system performs global registration of RGB-D scans in large indoor environments. Starting from a locally aligned input (left), it iteratively detects and enforces a structural model encoding hierarchical relationships between planar surfaces at larger and larger scales. In the final iteration (right), the global registration reproduces the global structure of the real-world scene.

when loop closures are not possible. For example in Figure 1, note how the walls of the corridor are straight and all corners are right angles in the registration results produced by our algorithm (right).

Though the main idea of our algorithm is intuitive, its implementation is not because there is a chicken-and-egg problem between global registration and structure detection. If the global registration is grossly inaccurate (as it often is in the early iterations of ICP), then it is difficult to detect large primitives and geometric relationships. If the large primitives cannot be detected robustly, then it is difficult to rectify the global registration. Neither problem can be solved without the other.

We address this joint problem with a novel fine-to-coarse ICP algorithm. During each iteration of the ICP algorithm, primitives are detected and constraints are added only within "windows" of sequential RGB-D images. The windows overlap and cover the entire sequence. They start small, become fewer and gradually increase in size as the ICP iterations proceed, ultimately leading to a single window containing the entire sequence in the final iteration. Because the global registration within each window is usually accurate enough to enable robust detection of primitives and their relationships at the scale defined by the window size, the constraints interjected into each ICP iteration are able to guide the algorithm towards a globally accurate solution. During experiments, we find that our fine-to-coarse ICP algorithm produces more accurate global registrations and handles more difficult inputs previous global registration algorithms.

Overall, the research contributions of this paper are three-fold.

- A system that integrates detection of geometric relationships (parallelism, perpendicularity, etc.) between hierarchical planar structures into the inner-loop of a global RGB-D registration algorithm.
- A fine-to-coarse iteration strategy that detects planar primitives, infers relationships, and finds correspondences only within gradually expanding windows with alignments optimized in previous iterations.
- A ground truth dataset containing 6,025 manually clicked point correspondences for evaluating global registration algorithms on SUN3D scans.
- An experimental study of how different components of the proposed approach affect global registration results with comparisons to alternative methods.

2 Related Work

There has been a long history of research on registration of RGB-D images in computer graphics, computer vision, augmented reality, robotics, and other fields [Sto16]. The following paragraphs describe the work most closely related to ours.

Real-time Simultaneous Localization and Mapping (SLAM): Most prior work has focused on real-time registration motivated by SLAM applications in robotics and augmented reality [Sto16]. Early systems used ICP to estimate pairwise alignments of adjacent video frames [BM92], feature matching techniques to detect and align loop closures [AFDM08], and graph-based optimization algorithms to perform a final global registration (e.g., [GKSB10]). More recent methods have improved robustness by aligning frames to a scene model, represented usually as a point cloud [HKH+10, KLL+13, RHHL02, WLSM+15] or an implicit function [CBI13, DNZ+16, KPR+15, NDI+11, WKF+12, WKJ+14]. However, since these model representations are unstructured, and alignments are often noisy, sequences of small local alignment errors can accumulate to form gross inconsistencies at large scales [NZIS13]. We address this issue by adding structure to the scene model.

Off-line Global Registration: To acquire the highest-quality final 3D model of a static environment from a RGB-D video, people usually use off-line global registration procedures. A common formulation is to minimizes an error function measuring misalignments between all overlapping pairs of frames [HKH$^+$10, ZK13, ZK14]. A major challenge in these approaches is to identify which pairs overlap – i.e., identify which pairs are true loop closures. Previous methods have searched for similar images with Bag of Words models [AFDM08], randomized fern encodings [WLSM$^+$15], convolutional neural networks [CLJM14], and other methods. A recent approach by [CZK15] proposed a method based on line processes that uses indicator variables to identify true loop closures during the global optimization using a least-squares formulation. They achieve impressive registration results for several scenes, but fail to preserve scene structure explicitly, do not scale well to larger environments, and fail completely for scans without loop closures (see Section 5).

Hierarchical Registration: Computing loop closures is especially difficult in areas with few salient features. So, some systems fuse sets of sequential frames into "chunks" and then treat the chunks as rigid bodies later in a global optimization [CZK15, TF15]. This approach adds robustness to loop closure detection (because chunks have more features to align than frames), and it saves computational resources (because there are fewer variables to optimize). Other methods fuse subgraphs of a loop closure graph hierarchically to improve optimization robustness and efficiency [ENT05, FLD05, RS15, TF15]. These methods share ideas with our fine-to-coarse strategy. However, our method extends them signficantly by integrating formation of new constraints (through a hierarchical structural model) based on the current alignment solution within the inner loops of the optimization.

Detecting Structural Features: Robust feature detection is an important component of almost every RGB-D registration system. Previous work has been based mainly on keypoints like SIFT [XOT13] and Harris corners [ZSN$^+$16]. However, other work has considered depth silhouette edges [ZK15], building structure lines [ZZP$^+$15], and planar regions [BS03, DGFF12, ERAB15, NHS07, PBVP, SMGKD14, TJRF13, TRC12, WS06] in order to improve the repeatability and distinctiveness of features. [MKSC16] associates planar features with global planes estimated with an E-M algorithm. We build upon this trend towards higher-order structural features by detecting planar regions *and relationships between them* in the inner loop of a fine-to-coarse registration algorithm that is robust to poor initial alignments.

Manhattan World Reconstruction: Many man-made enviroments are composed of large planar features aligned with orthogonal axes. This Manhattan World assumtion has been exploited in previous 3D reconstruction systems. For example, [SB08] reconstructs 2D floorplans by assigning the local orientation of every 2D line to exactly one of two global orthogonal directions based on its initial alignment. [FCSS09] uses the Manhattan World assumption for dense depth estimation in RGB images they propose to cluster pixel normals into three directions to improve the pairwise term for a CRF. While these papers make the same assumption we do, they do not share any of our goals, insights, algorithms, or results.

Extracting Structural Models: There has been concurrent work on extracting structural models from RGB-D scans of objects and scenes. All of this work assumes that the RGB-D scans have already been approximately registered, which makes detection of the primitives relatively easy with methods like RANSAC [OVWK16, SDK09, VLA15, WKJ$^+$14, WACS12]. For example, GlobFit detected a set of primitives with RANSAC to form an initial model and then optimized their fits by detecting and enforcing geometric constraints between them [LWC$^+$11]. RAPter extended that work by integrating primitive and constraint detection into a single optimization [MMBM15]. [NRS15] extended it even further by integrating slight optimizations of camera poses into the process. We draw upon many ideas in these papers, including the RAP (regular arrangement of planes) representation encoding local planar regions and global inter-plane relationships from RAPter [MMBM15]. However, we extend them to the more common case in which RGB-D images are not registered in advance. Global reconstruction of long RGB-D sequences in complex indoor environments is very difficult, and so detection and enforcement of structural constraints is more important in that early stage of the RGB-D processing pipeline than during beautification at the end. Investigating that idea and its implementation is the main contribution of this paper.

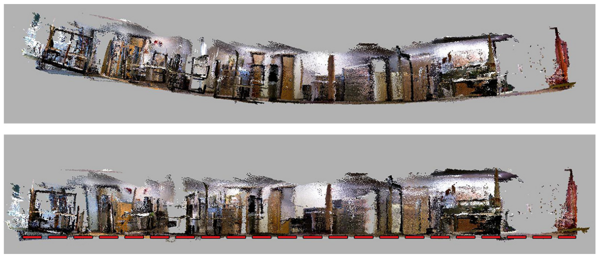


Figure 2: Fitting and enforcing a structural model comprising planes with long-range constraints can yield more accurate global registration. The top image shows our registration result without detecting and enforcing a structural model, and the bottom one shows our result. The horizontal red line depicts the planar proxy found in the later iterations of our algorithm that guides the optimization to the correct solution.

3 Approach

In this paper, we describe an algorithm that leverages detection and enforcement of a structural model to assist robust and accurate global registration of RGB-D video frames.

The core idea that makes our algorithm unique is that it utilizes discovery and fitting of a structural model within the inner-loop of an iterative alignment optimization. The algorithm follows the general E-M strategy of ICP: alternating between a discrete E step (extracting a structural model and establishing constraints) and a continuous M step (solving for the camera poses that best satisfy the constraints). The core difference is in the E step: we detect new structural features spanning multiple frames and establish long-range geometric constraints between them based on the current transformations at each iteration.

The benefits of this idea are self-evident: detected relationships between large-scale structural features spanning multiple frames provide valuable constraints for global registration. For example, Figure 2 shows a simple case where a structural model incorporating coplanarity constraints between sections of floor along a long corridor provide critical cues for accurate global reconstruction. Constraints of this type are quite common in indoor environments, where architecture generally adheres to the Manhattan World assumption. So, we focus our study on structural models comprising hierarchical sets of planes with geometric constraints between them (coplanarity, parallelism, orthogonality). With rare exceptions, we find that this structural model is flexible enough to fit a wide variety of indoor environments, but constrained enough to guide our algorithm towards correct registrations.

However, the implementation of this core idea is not easy, since large-scale planar features spanning multiple frames cannot be detected robustly unless the frames have already been aligned. Unlike prior systems for structural modeling of indoor scenes (e.g., [LWC+11, MMBM15, NRS15]), we do not assume that the RGB-D scans are initially in nearly correct global alignments – achieving that is actually the most difficult part of processing long scans over large areas. We aim for a method to detect and rectify structure in scans that exhibit large-scale drifts leading to gross global misalignments.

To address this issue, we introduce a fine-to-coarse iterative algorithm for simultaneous structure detection and camera pose estimation. The algorithm alternates between a) detecting structural constraints within overlapping "windows" of sequential RGB-D frames based on their current camera poses estimates, and b) optimizing the camera pose estimates to satisfy the detected constraints.

The algorithm is called "fine-to-coarse" because the windows start small and get larger as the algorithm proceeds. In the early iterations, the windows span just a few sequential RGB-D frames where relative camera poses estimated with a local tracking algorithm should be nearly correct. At this early stage, for example, it should be possible to detect coplanarity and orthogonality constraints between nearby surfaces in adjacent frames (see Figure 3b). As the iterations proceed, the windows get larger, enabling detection and enforcement of larger-scale and longer-range structural constraints (Figure 3c). At each iteration, we expect the relative camera poses within each window to be approximately correct, since they have been optimized according to constraints detected in smaller, overlapping windows in previous iterations. Thus, we expect that it will be possible to detect the relevant structural constraints within each window robustly based on the current camera pose estimates. Ultimately, in the final iteration, the algorithm uses a single window encompassing the entire scan. At that stage, it detects and enforces a single structural model within a global optimization of all camera poses (Figure 3d).

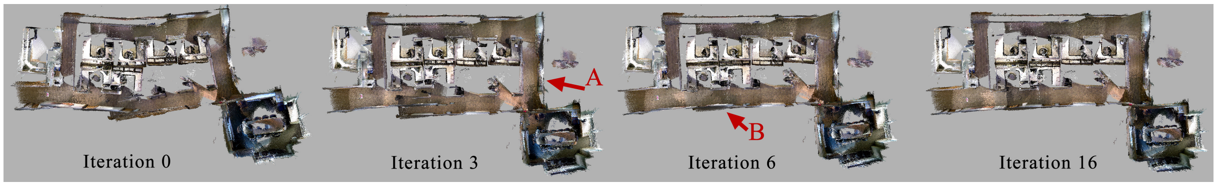


Figure 3: Fine-to-coarse registration. Starting with an initial alignment T_0 shown on the left, our algorithm detects and enforces structures at small scales in the first few iterations (straightening the wall marked 'A' by the 3rd iteration), then detects and fixes longer-range structural misalignments in the middle iterations (aligning the walls marked 'B' by the 6th iteration), and finally snaps everything together into a rectified global registration by the final 16th iteration.

4 Methods

The input to our system is a sequence of n RGB-D images I acquired with a hand-held camera with measured intrinsic parameter, and the outputs are 1) a set of rigid transformations for all images, 2) a structural model S representing planar structures and geometric relationships between them, and 3) a set of aligned surfels with positions, normals, colors, and radii.[1]

As shown in Figure 4, our algorithm starts by executing three "preprocessing" steps. First, for each input image $I[j]$ ($j \in (1, n)$) in the input scan, we extract a dense set of 3D features F. Second, we estimate a local transformation $L[j]$ between each pair of consecutive images $I[j]$ and $I[j+1]$ and concatenate them to obtain an initial set of image transformations T_0. Third, we initialize a structural model $S_0 = \{P, H = \emptyset, G = \emptyset\}$, where P is a set of planar proxies representing sets of connected coplanar pixels in each image, H is the set of hierarchical structure of coplanar proxies, and G is a set of geometrical relationships between the members of H.

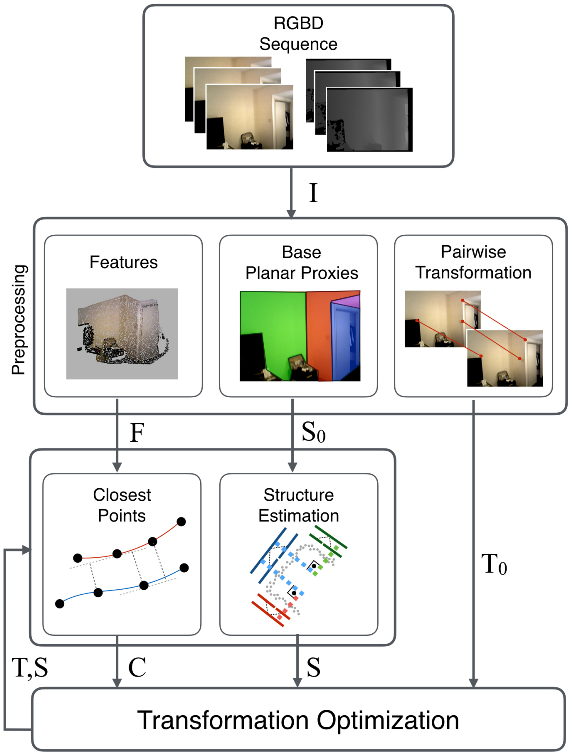


Figure 4: Flow of data in our system.

We then proceed to iteratively refine the image transformations T and structural model S with the proposed fine-to-coarse strategy. In each iteration i, we choose a window size $w[i]$ and process a set of overlapping windows of $w[i]$ consecutive images, where adjacent windows overlap by a factor of 50%. For each window $W[i][j]$, we: 1) cluster coplanar proxies from multiple images within $W[i][j]$ to form new larger-scale proxies and add them to P; 2) insert parent-child relationships between newly created proxies and members of its cluster into H, 3) detect geometric relationships between newly created proxies in the same and adjacent windows and add them to G; and 4) find closest point correspondences C between features F from pairs of images in $W[i][j]$. We then refine rigid motion parameters of the transformations T and proxies P to minimize an error function that is a weighted sum of terms penalizing deformations of structural relationships (E_S), distances between corresponding features (E_C), misalignments of local transformations (E_L), and large changes to the solution (E_I):

$$\begin{aligned} E(T,S,C) =& w_S E_S(S) + w_C E_C(T,C) \\ & w_L E_L(T) + w_I E_I(T) \end{aligned} \tag{1}$$

We set the weights for the error terms dynamically as the algorithm iterates. The early iterations have higher weights for error term enforcing struc-

[1] The aligned surfels output by our system can be used by any algorithm to reconstruct a continuous surface, but we do not consider that part of our algorithm – i.e., we focus on scan registration, not surface reconstruction.

ture (w_S), and the later ones have higher weights for closest point correspondences (w_C), with a gradual blend of weights during the middle iterations. Intuitively we want early iterations to roughly align the large scale structures, and then let closest point correspondences take over and align surfaces precisely.

Descriptions of these error terms, their computation, and the optimization appear in the following subsections (and appendix). Though each subsection has limited novelty on its own, the system as a whole is quite novel. Our main research contribution is the combination of the following steps into a fine-to-coarse algorithm that performs simultaneous structure estimation and global registration.

4.1 Feature Detection

The first step of our process is to extract a dense set of features F from the input depth images. These features will form the basis for associating images to structures and for finding closest point correspondences. Thus, we aim for a dense, but well-spaced, set of features capturing salient planar and edge properties of every depth image.

The image processing steps for feature detection are described in the appendix. We experimented with a variety of feature types, including corner points and ridge/valley lines, but found that most were too noisy to be helpful during our experiments. So, we converged on a set of features comprising points on silhouette edges, planar regions, and uniform samples. Every feature is represented by a 3D position, direction, salience, and radius, where the direction represents a line vector for features on silhouette edges and a normal vector for others.

Since storing and searching a set of features for every pixel in every images would be impractical for a long RGB-D scan, we subsample features with a Poisson dart algorithm. We first visit pixels on silhouette edges of the depth image, next on planar proxies, and finally all pixels, inserting features into F if and only if they are at least a minimum spacing from any previously generated feature (*min_spacing*=5cm). The net result is a set of features for every image that densely covers all salient surfaces, but is not too large to be overwhelming for subsequent steps of the algorithm. [2]

4.2 Adjacent Frame Registration

The second step of our process is to compute the relative rigid transformation $L[j]$ between each pair of consecutive images $I[j]$ and $I[j+1]$. These transformations will be concatenated to provide an initial set of transformations for all images T_0, and they will be used to compute an error term E_L that favor trajectories matching the predicted local transformations between nearby frames.

A possible way of estimating transformations for adjacent viewpoints is to use frame-to-model pose estimation, as presented in [NDI+11], [WKF+12] or [KLL+13]. While those techniques are quite successful for scans of objects and/or densely populated regions of a room, they are not as effective in large-scale interior environments – i.e., they lose tracking when cameras travel down long corridors, pan flat walls, and/or visit regions of the scene without distinctive depth features.

Instead, we decided to leverage matching of color features for alignment of adjacent frames. Specifically, using the method described in [XOT13], we detect SIFT features in color images that have valid depths, use RANSAC to find 3D feature correspondences (x_k, x'_k), and solve for $L[j]$ by minimizing $\sum_{k=1}^{K}(x_k - L[j](x'_k)^2$.

The resulting pairwise transformations $L[j]$ are then concatenated to form an initial transformation for every image in a world coordinate system ($T_0[0] = I$, $T_0[j] = L[j-1](T_0[j-1])$ for $j \in [1, n-1]$).

Clearly these initial transformations suffer from a lot of drift. However they are usually correct locally, and thus we can use them to compute an error term E_L to favor preserving local relative transformations between nearby images as:

$$E_L(T) = \sum_{j=0}^{n-1} \sum_{k=0}^{kmax}$$

$$E_t(T_0[j+2^k]^{-1}(T_0[j]), T[j+2^k]^{-1}(T[j]))$$

where *kmax*=4 and E_t measures the misalignment of two transformations as described in the appendix.

4.3 Structural Model Initialization

The third step of our process is to initialize the structural model with a set of planar proxies. These proxies are extracted from the depth channel of each input image independently. Hierarchical and geometric relationships between them will be inferred

[2] These subsampled features are the ones drawn in all renderings of results in this paper.

during the iterative refinement steps described in the next section.

To extract planes from depth images, we investigated a variety of standard algorithms, including region growing and RANSAC as implemented in the Point Cloud Library [RC11], but ultimately chose to implement our own agglomerative hierarchical clustering algorithm, which we found worked better on noisy data sets, like SUN3D. In any case, the result is a set of planar proxies P, each associated with an image and represented by a centroid, normal, radius, and set of associated planar features.

For each proxy $P[j]$, we establish a set of coplanarity relationships between the proxy and a sampling of m of its associated features as:

$$E_P(T, P) = \sum_{j=1}^{|P|} \sum_{k=1}^{m} E_{cp}(P[j], T[ik](F[k]))$$

where ik is the index of the image containing feature $F[k]$, $E_{cp}(A, B)$ measures the deviation of two planar structures from being coplanar as described in the appendix, and m is a number proportional to inliers of proxy $P[j]$, selected such that $\sum m_j = 100n$

4.4 Structural Model Refinement

The next step of the process is to update the structural model based on the current transformations. The inputs to this step at each iteration are the structural model S and transformations T computed in the previous iteration, and the output is an augmented structural model S containing new proxies and relationships between them, plus error terms that favor adherence to the structural model when solving for image transformations:

$$E_S(T, S) = w_P E_P(T, P) + w_H E_H(S) + w_g E_G(S)$$

where the three subterms account for fits of planar proxies to depth images (E_P, w_P=2000), coplanarity of hierarchical proxy relationships (E_H, w_H=2000), and detected geometric relationships between proxies (E_G, w_G=2000), as described in the following subsubsections.

Planar Proxy Extraction: The algorithm begins by grouping parentless planar proxies from the previous iterations within the given window $W[i][j]$ into coplanar sets represented by new proxies in the current iteration.

Our algorithm performs this grouping with a hierarchical clustering strategy. At the start, a new proxy is created for each parentless proxy from the previous iteration. Then new proxies are iteratively merged in order of highest affinity until no further merges are possible, where affinities are computed between two proxies with a function summing the D minus the distance between one proxy and the centroid of the other and A minus the angle between proxy normals. Exceeding D=10cm or $A = \pi/4$ terminates the merging process. The final result is a small set of proxies associated with co-planar proxies from the previous iteration.

For each new proxy, we define a parent-child coplanarity relationships with its associated proxies from the previous iteration. After a few iterations, we end up with a hierarchical set of planar proxies linked by coplanarity relationships (Figure 5a). The resulting structure has $O(klogk)$ constraints between k coplanar base proxies, which is far fewer than $O(k^2)$ that would be required without any hierarchy, and therefore more efficient. However, it also provides some localized structure and redundancy as compared to creating $O(k)$ constraints linking all off them to one master proxy, and therefore is more robust to mistakes.

Figure 5a depicts an example of how planar proxies are extracted across three iterations (each color represents a different iteration). In the schematic image, input depth pixels are shown as dots, proxies are shown as straight solid lines, and parent-child coplanarity constraints are shown as dotted lines. For ease of visual interpretation, proxies extracted in later iterations are offset further from the input depth images (like an exploding view diagram). This figure depicts how earlier iterations form proxies covering smaller windows and how multiple iterations form a hierarchy of proxies linked by coplanarity relationships. Please note that proxies overlap by 50% (not shown in the schematic), and so each proxy from one iteration can be associated with more than one proxy in the next.

The coplanarity constraints of the hierarchical parent-child relationships between proxies P[j] and P[k] are enforced with an error term:

$$E_H(P) = \sum_{j=1}^{|P|} E_{cp}(P[j], P[k])$$

where $E_{cp}(A, B)$ measures the deviation of two planar structures from being coplanar as described in the appendix.

Structural Relationship Detection: Once the planar proxies are detected, we search for salient ge-

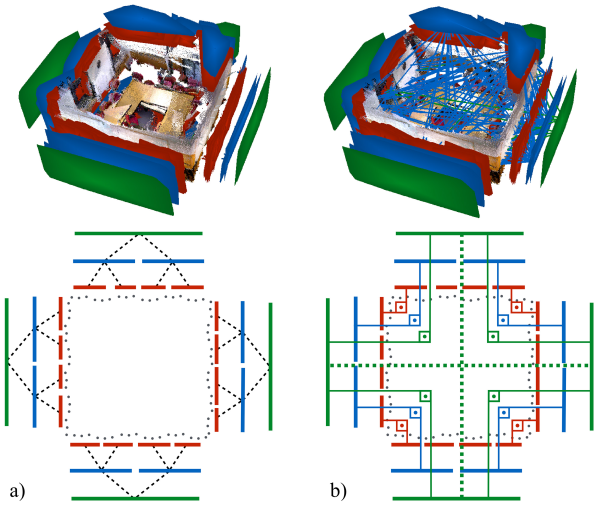


Figure 5: Visualization of a structural model (top row) with a schematic representation (bottom row). a) Planar proxies detected in individual depth images (solid red lines) are linked via hierarchical relationships through fine-to-coarse iterations (blue and green). b) Planar proxies created in the same iteration are linked by geometric relationships representing antiparallelism (dotted green line) and orthogonality (thin solid lines). These relationships provide error terms to guide the global registration.

ometric relationships between ones within the same or adjacent windows. The goal here is to detect relationships that are nearly perfect (nearly parallel, nearly perpendicular, and nearly coincident) and then create weighted constraints between them that encourage rectifying imperfections during the global optimization.

Since the number of proxies considered in each iteration is quite small, and the relative alignments within each window are nearly correct as a result of optimizations of transformations in the previous iterations, the algorithm for detecting geometric relationships between proxies can be quite simple. We consider all pairs of proxies. For each pair ($P[a] = \{\mathbf{n}_a, q_a\}$ and $P[b] = \{\mathbf{n}_b, q_b\}$) within the window, we create a typed structural relationship $S[a][b]$ with weight $w[a][b]$. If the angle $\theta_{ab} = acos(\mathbf{n}_a \cdot \mathbf{n}_b)$ is less than $\frac{\pi}{4}$, we create a parallel relationship with weight $w[a][b] = e((\theta_{ab} - \pi)/\sigma_\theta)$. If the $\theta_{ab} > \frac{3\pi}{4}$, we create an anti-parallel relationship with weight $w[a][b] = e(-\theta_{ab}/\sigma_\theta)$. Otherwise, we create a perpendicular relationship with weight $w[a][b] = e(-(1 - \theta_{ab})/\sigma_\theta)$. For parallel relationships, we mark them as coplanar if the distance from the plane of each proxy to the centroid of the other is less than σ_{dist}.

Figure 5b depicts an example of structural relationships extracted across three iterations. Perpendicular relationships are shown as thin-lines, and antiparallel relationships are shown as dotted lines (coplanarity relationships not shown). Please notice that structural relationships are formed only between proxies at the same iteration level, forming a hierarchy of structural relationships matching the hierarchy of proxies. Please also note that long range relationships (e.g., the antiparallel green proxies) are detected only in the later iterations of the algorithm, when proxies have had a chance to grow and converge. This feature of the fine-to-coarse algorithm provides robustness to initial misalignments.

For each detected geometric relationship g between proxies $P[j]$ and $P[k]$ with transformed normals n_j and n_k, respectively, we define an error term:

$$E_g(g) = \begin{cases} w[j][k](P[j](n_j) - P[k](n_k))^2 & \textit{parallel} \\ w[j][k](P[j](n_j) + P[k](n_k))^2 & \textit{antiparallel} \\ w[j][k](n_j \cdot n_k)^2 & \textit{orthogonal} \\ w[j][k]E_{cp}(P[j], P[k]) & \textit{coplanar} \end{cases}$$

These errors are summed over all geometric relationships constructed in all iterations to form an aggregate error:

$$E_G(G) = \sum_{j=1}^{i} \sum_{j=1}^{|G|} E_g(G[j])$$

4.5 Closest Point Correspondences

During each iteration, we also construct a set of closest point correspondences that help “snap” scans together precisely across loop-closures. In most ways, this step is like any other correspondence finding operation in a global ICP optimization – for every pair of scans, we construction correspondences from features of one to the closest compatible feature of the other. However, we make a few adjustments from the usual algorithms to account for the specifics of our setting.

First, we form closest point correspondences only between features within the same window $W[i][j]$ of frames at each iteration. Closest point correspondences are notoriously sensitive to initial transformations, and the transformations at the start of each iteration of our algorithm have been optimized only within windows during the previous iterations. So, it is safest to construct correspondences between closest points on different images only if they are in the same window. This is implemented very simply by considering pairs of images only if they are separated by less than $w[i]$ when forming closest point correspondences.

Second, we utilize a new method for rejecting outlier correspondence that depends on the window size and the difference between image indices in the scan. Other methods often utilize a single distance threshold (*max_distance*) for rejecting outlier correspondences. They may compute the threshold from statistics of previously computed closest point distances and/or reduce the distance in later iterations [RL01]. However, they still use a single threshold for all features in the same iteration because they expect the registration error to be approximately uniform across the scan. In contrast, since we use a fine-to-coarse iterative strategy, we do not expect the distance between inlier correspondences to be uniform for all pairs of images. Features from images nearby in the scan will have had a chance to form correspondences several times in early iterations when windows are small before features from distant pairs of images are considered for correspondence even once. Thus, we apply a different threshold for the maximum distance between inlier feature correspondences for different features based on the window size of the current iteration and the difference between image indices in the scan.

Third, we consider closest point correspondences for all pairs of images within every window in one large global optimization. This is in contrast to methods that use ICP to compute pairwise transformations and then optimize transformations to match them [GKSB10] and to methods that perform global ICP for small sets of range scans [LSP08]. The advantage of the global approach is that joint optimization of all constraints together helps converge more robustly to the global optimum. The disadvantage is that computing and storing correspondences for long scans is onerous. Although a kd-tree is used to accelerate closest point searches, performing searches for every feature of every image to find closest points on every other image is impractical (a typical scan has 10^3 features, and a typical RGB-D video has $10^3 - 10^4$ frames). To overcome this issue, we have developed a randomized algorithm to select features to use for closest point searches at each iteration. We first select a maximum number of closest point searches c (in our implementation $c = 100n$). Then, for every pair of images i and j, we compute the fraction of c proportional to the volume of their bounding box intersections and search with that many features from image i to find closest compatible features in image j. We select features from image i with probability proportional to their salience. For each of those features, we find the closest compatible feature (within the dynamically computed *max_distance* and *max_angle*=$\pi/4$) [Pul99], discarding all correspondences whose closest point is a boundary in its image [TL94].

The net result of this process is a set of feature correspondences C containing pairs of salient features spread nearly uniformly throughout the scan sequence. Each one-way correspondence between features $F[a]$ and $F[b]$ provides a constraint represented by an error term as follows, where p_a and p_b are the transformed positions of features $F[a]$ and $F[b]$, and n_a and n_b are the transformed normals of features $F[a]$ and $F[b]$, respectively.

$$E_c^{\rightarrow}(T, F[a], F[b]) = \begin{cases} ((p_b - p_a) \times n_a)^2 & linear features \\ ((p_b - p_a) \cdot n_a)^2 & planar features \end{cases}$$

A symmetric error is computed for every correspondence c associating features $F[a]$ and $F[b]$ as:

$$E_c(T, c) = E_c^{\rightarrow}(T, F[a], F[b]) + E_{-c}^{\rightarrow}(T, F[b], F[a])$$

These errors are summed over all correspondences C created in the current iteration to form an overall correspondence error:

$$E_C(T, C) = \sum_{j=1}^{|C|} E_c(T, C[j])$$

4.6 Optimization

At the end of each iteration, we perform an optimization to solve for new rigid transformations for every image T and every proxy P.

The objective function contains a weighted combination of the error terms described in the previous subsections (Equation 1). Additionally, we include an inertia term to encourage small changes at each step:

$$E_I(T, P) = \sum_{j=1}^{|I|} (\Delta T[j])^2 + \sum_{j=1}^{|P|} (\Delta P[j])^2$$

where ΔA represents the sum of squared differences between Euler angle rotations and translations for A from one iteration to the next. This inertia term not only provides stability for the optimization, by damping changes, but it also is required to prevent the system of equations from being underconstrained.

Although the error function is non-linear, we find that our overall algorithm converges more quickly when the variables are relaxed with a linear approximation once per iteration (e.g., using CSparse[Dav06]). Though this approach is related to solving the non-linear function with a series of linear approximations (e.g., as is common in solvers like Ceres [AMO]), there is a significant difference – we update the structure of the error function between solving every linear approximation (by finding new structural and closest point constraints). Since updating the error function is faster than solving the linear approximation, and since the solution to the updated function is generally closer to the final solution, the overall convergence of our algorithm is faster if only one linear approximation is solved per iteration.

5 Evaluation

We have executed a series of experiments designed to test the performance of our proposed method. The following describes the datasets, evaluation metrics, comparisons, and findings of these experiments.

Datasets and Evaluation Metrics: Datasets with RGB-D scans of large environments are just now becoming available. Traditional RGB-D datasets are targeted at surface reconstruction and/or camera tracking over small ranges, and thus mostly contain scans covering (parts of) a single room [AKJS12, DNZ$^+$16, HWMD14, LBF14, MPM$^+$14, SHKF12, SEE$^+$12]. However, we target scanning of large-scale interior environments, like an apartment or museum.

The most commonly used public RGB-D dataset of that type is SUN3D [XOT13]. It contains a set of 415 RGB-D videos captured with a ASUS Xtion PRO LIVE sensor attached to a hand-held laptop in 245 distinct spaces (apartments, hotel rooms, classrooms, etc.). Each scan contains $10^3 - 10^4$ images, sometimes covering multiple rooms. Plus, images are low-resolution and noisy. So, they provide a challenging dataset for our work.

However, SUN3D does not provide ground truth camera poses or even correct global registrations (except for 8 scenes aligned approximately with object bounding box correspondences). So, we were compelled to create ground truth alignment data for evaluation of our method. We selected 24 scenes, of which 8 are the ones approximately aligned manually by [XOT13] and previously studied in [CZK15], plus 16 more selected because they appear to be among the most interesting examples in the dataset (we call this the SUN3D test set).

For these 24 test cases, we manually specified a total of 6,025 ground truth point correspondences with an interactive visualization tool (97-645 per scan). The correspondences are concentrated in global loop closures, but also contain pairs of points spanning nearby frames spread evenly through the scan (as shown in the supplemental material). We measure the accuracy of an alignment by the root mean squared distance (RMSD) between all pairs in these ground truth correspondence sets.

Alternative Methods: We use this new SUN3D test set to evaluate how accurately our proposed system aligns RGB-D scans of interior environments and compare to alternative methods proposed in previous work.

We focus our comparisons on the following two off-line global registration algorithms. These methods were chosen because: 1) they provide the best previous results for our target dataset (SUN3D) [CZK15], 2) they take very different approaches to the global registration problem, and 3) they can be executed with code provided directly by the authors (we used that code with default parameters to produce results for all experiments). We would have liked to compare to [DNZ$^+$16], but their paper has been on arXiv for only one month and their code is not yet available.

- **Robust Reconstruction [CZK15]:** This method fuses adjacent sequences of 50 frames into fragments, aligns all pairs of fragments with a variant of RANSAC using FPFH features extracted from depth images, selects pairs as potential loop closures, and then solves a least squares system of nonlinear equations that simultaneously solves for camera poses and loop closure weights. We believe this method is the state-of-the-art for off-line, global registration using only depth images.
- **SUN3DSfM [XOT13]:** This method aligns pairs of frames with a RANSAC algorithm based on SIFT features extracted from RGB images and then solves for a global alignment with a joint 2D+3D bundle adjustment. The solver includes error terms for all consecutive pairs of frames, plus expected loop closures found with a Bag of Words model and verified to

have significant confidence. This method is representative of the state-of-the-art for off-line, global registration using mainly RGB features.

For completeness, the supplemental material also contains (unfair) comparisons to ElasticFusion [WLSM$^+$15] and Kintinuous [WKF$^+$12]. They do not perform as well on our experiments, which is understandable since they are designed to run in real-time robotic applications, while ours is designed for off-line scene capture applications.

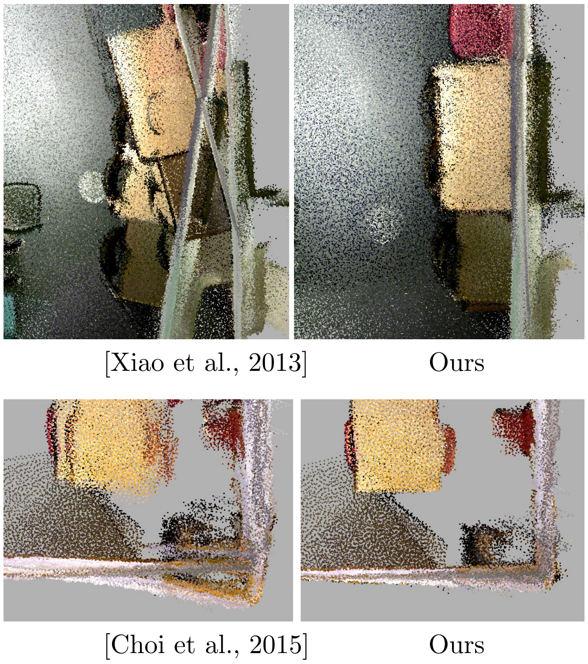
Qualitative Comparisons: Figure 6 shows a visual comparison of results for several interesting test cases from the SUN3D dataset (comparisons for all 24 test cases appear in the supplemental material).

Looking at these results, it is quite apparent that fitting a structural model helps to guide our algorithm (right column) towards a qualitatively correct result for these test cases. Generally speaking, the corners of rooms appear as right angles, opposite walls appear parallel, corridors appear straight, floors are flat, etc.

In contrast, [XOT13] and [CZK15] produce results with obvious warps and misalignments. For example, [XOT13] achieves good local alignments, but almost always generates large-scale drifts. Though they search for loop closures with a Bag of Words approach, they do not always find the right closures (left side of Figure 8a).

[CZK15] produces good results for single rooms with loop closures (we are able to duplicate the results from their paper). However, they do not always reproduce the correct Manhattan structure (left side of Figure 8b), and they seem to perform poorly for longer RGB-D sequences in larger environments. [CZK15] was never tested for large scenes, and so it is difficult to know if the latter problem is only due to parameter settings. However, it appears to find incorrect loop closures that cause gross misalignments. For example, in the third row of Figure 6, it merges the two rooms into one presumably because of incorrect matches between geometric elements repeated in the different rooms. In this case and others, there are few loops in the camera trajectory, and the geometry in revisited areas is not distinct, and so methods based on loop closures are bound to fail. Our method can succeed even when there are no loop closures at all.

Interestingly, our method works even in cases where the scene does not adhere to the Manhattan World assumption. For example, it correctly reproduces the non-rectangular shape of d507_2. Because the weights assigned to off-angle geometric relationships are quite low in comparison to closest point correspondences in the later iterations, our algorithm converges to the correct solution.



[Xiao et al., 2013] Ours

[Choi et al., 2015] Ours

Figure 8: Example problems addressed by our method.

Quantitative Comparisons: Figure 7a shows quantitative results for our method in comparison to alternatives on the SUN3D dataset. For each scene listed along the horizontal axis, there are three bars representing the RMSD error of ground truth correspondences for [XOT13] (red), [CZK15] (green), and our algorithm (blue). From this plot, it can be seen that our reconstructions align the ground truth correspondences better than either of the other two methods. Our RMSD is better (shorter bar) in 15/24 test cases, and nearly equivalent in the other 5. Our average RMSD error is 10.8cm, which is 3.7X less than [XOT13] (40.1cm) and 9.0X less than [CZK15] (97.1cm). Our maximum RMSD error is 25.7cm, which is 6.6X and 21.6X less than the others. These results (and the more detailed information in the supplemental material) suggest that our method out-performs the previous state-of-the-art on global registration of scans in multi-room indoor environments.

Evaluating Algorithmic Contributions: We ran an additional experiment to test the impact of the two core ideas of this paper: structural modeling and fine-to-coarse refinement. The goal

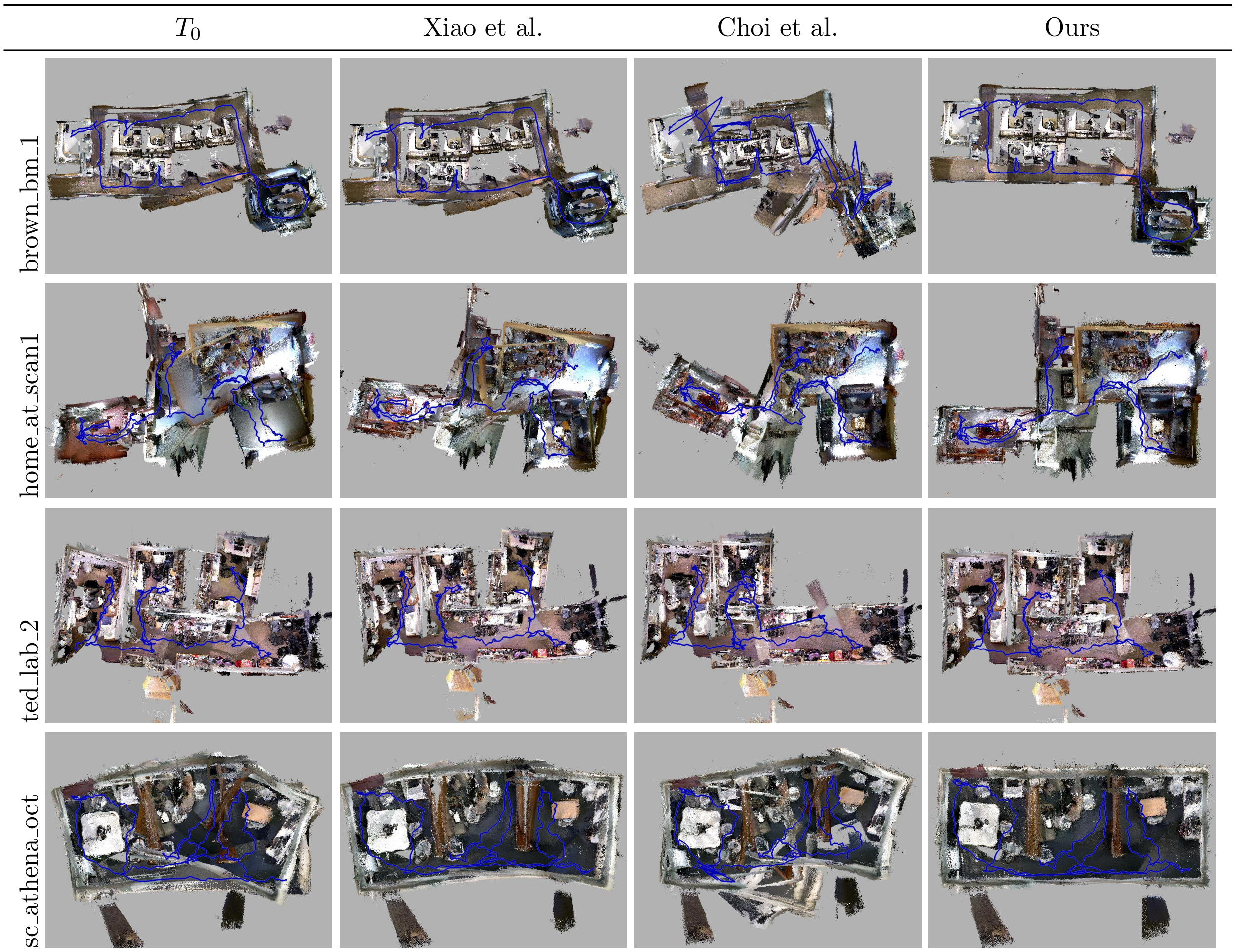


Figure 6: Qualitative comparison of global registration results for some examples. The rightmost column shows our results. The leftmost column shows the solution used to initialize our algorithm (T_0). The middle two columns show results produced with alternative algorithms from the literature. Note that images were created by rendering points spaced by at least 5cm in every 5th image for pragmatic efficiency.

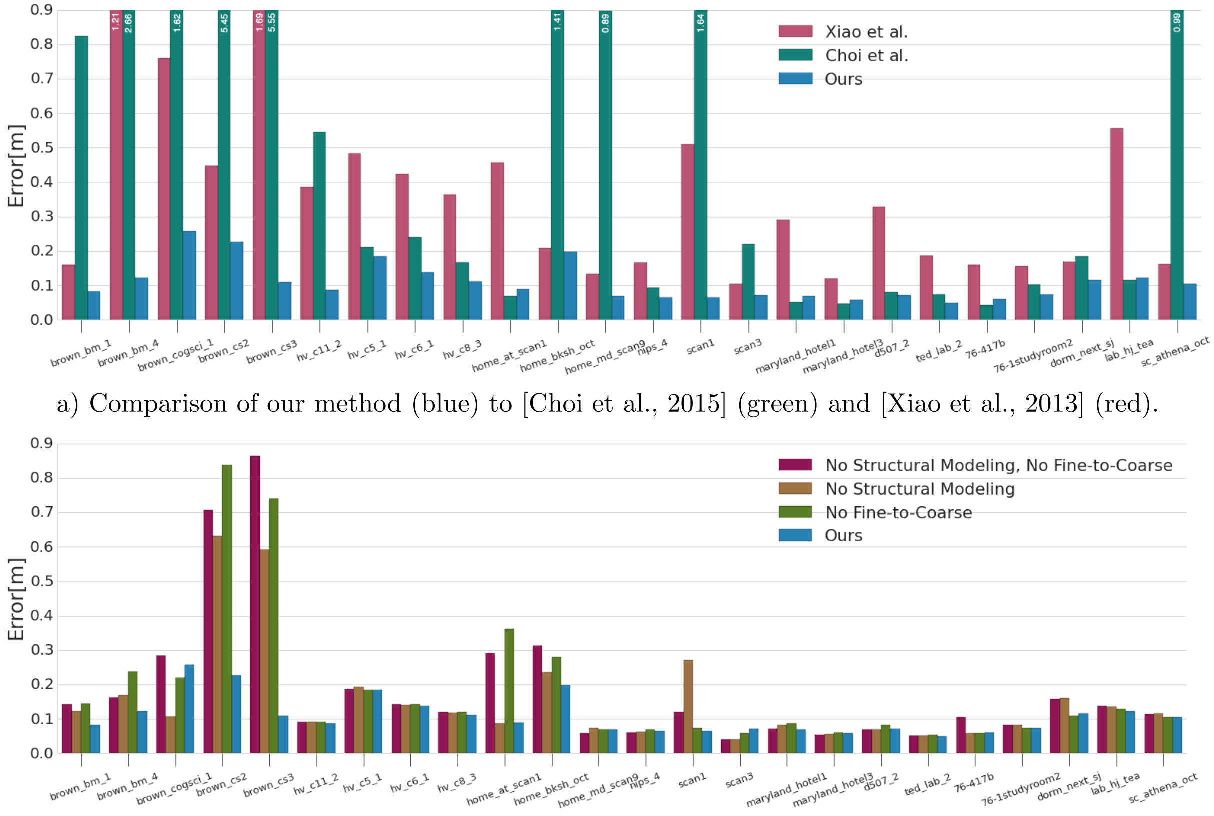
is to quantify how much these ideas contribute to our final results.

For this experiment, we run four executions of our system, one for each combination enabling/disabling structural modeling and/or fine-to-coarse refinement. When structural modeling is disabled: $w_S = 0$. When fine-to-coarse refinement is disabled: $w_i = n$. We run each variant of the system on all 24 examples in the SUN3D test set and compare the RMSD errors of the results.

Figure 7b shows the results with a bar plot with the same axes as Figure 7a. Again, the bars representing our proposed algorithm are blue. The three to the left of them for each test scene, show the RMSD for the variants of our algorithm with different core ideas disabled.

From these plots, we see that structural modeling improves the results for most test scenes, though some more than others (blue bars are almost always shorter than yellow bars). For simple scenes with lots of trackable features (e.g., 76-1studyroom2), our algorithm performs well without a structural model, and so there is no significant difference. However, for large scenes with few loop closures (e.g., brown_cs_2, brown_cs_3, etc.), there are big differences (1.5-5.0X improvements). On the whole, we conclude that enabling structural modeling almost always provides a noticeable improvement for the difficult cases, and rarely diminishes results for the easy cases.

We see a similar trend when studying the impact of fine-to-coarse alignment. It it essential for the long scans (e.g., the ones listed above, plus brown_bm_1, brown_bm_4, etc.), where the initial transformations exhibit lots of drift. However, it is not that helpful in small scenes with lots of surface features where concatenating local alignments is sufficient to provide an approximate global alignment. Again, we conclude that enabling fine-to-coarse alignment almost always provides a noticeable improvement for the difficult cases, and rarely diminishes results for the easy cases.



a) Comparison of our method (blue) to [Choi et al., 2015] (green) and [Xiao et al., 2013] (red).

b) Comparison of our method (blue) to versions without fine-to-coarse alignment (green), structural modeling (yellow), and both (red).

Figure 7: Comparison of RMSD errors for ground truth correspondences of the SUN3D test set.

Failure Cases: Our algorithm makes errors for some types of inputs. For example, in Figure 9, the structure is estimated correctly, but still our result is incorrect (the two rooms at the top are too close to one another). In this case, the only constraints preventing the two rooms from translating horizontally are local alignments of features in the hallway across the bottom. Since there are few of those, our optimization incorrectly slides the camera poses parallel to the planes detected for the hallway walls. Better estimation of local transformations and/or extraction of finer-scale features might improve failures of this type.

Timing Results: Our algorithm is intended for off-line use and thus has not been optimized or coded for a GPU. After preprocessing steps to detect planes, perform alignment of adjacent images, and sample depth image features, our core algorithm (fine-to-coarse iterations of structural model refinement, closest point correspondence, and optimization) takes 36 minutes on average for the 24 examples in the SUN3D test set when executed on a single CPU running Linux (the minimum time was 4 minutes and the maximum was 136 minutes). These compute times are competitive with previous work and well within practical limits, as they are incurred once in the lifetime of each scan.

6 Conclusion and Future Work

This paper describes a method for off-line global registration of RGB-D scans for typical indoor environments. The key idea is to integrate detection and enforcement of a structural model into the inner-loop of a global registration algorithm. By executing the registration with a fine-to-coarse iterative strategy, we find that it is possible to detect a multi-resolution structural model while the registration is underway. Planar primitives and relationships between them detected as the scene is registered provide constraints that guide the algorithm towards accurate reconstructions in typical indoor scenes. Results of experiments demonstrate that the proposed method outperforms previous work on benchmark datasets, including a new one created with SUN3D data that will be made publicly available.

This work is a first step into a large space of potential algorithms combining shape analysis and global registration, and thus there are vast opportunities for future related work. First, we use RGB channels only to register adjacent frames in our framework – clearly there are opportunities to get much better alignments by utilizing RGB features to detect loop closures, for example as they become proximal during the fine-to-coarse alignment. Second, our structural model focuses on Manhattan Worlds – it will be interesting to explore which types of structural models work best for other types of environments. Finally, our study focuses only on combining shape analysis and global registration – integrating surface reconstruction into the inner loop of the pipeline is a compelling avenue for future study.

Appendix

This appendix contains low-level implementation details.

Exracting planar proxies from images: The following sequence of operations is performed on the depth channel for each image $I[j]$ to extract an initial set of proxies (see Section 4.3):

1. Apply a bilateral filter to reduce noise and quantization effects ($\sigma_{xy} = 3pixels$, $\sigma_{depth} = 5cm$).
2. Mark pixels on silhouette/shadow boundaries if their depths differ from any of their neighbors by more than 10%.

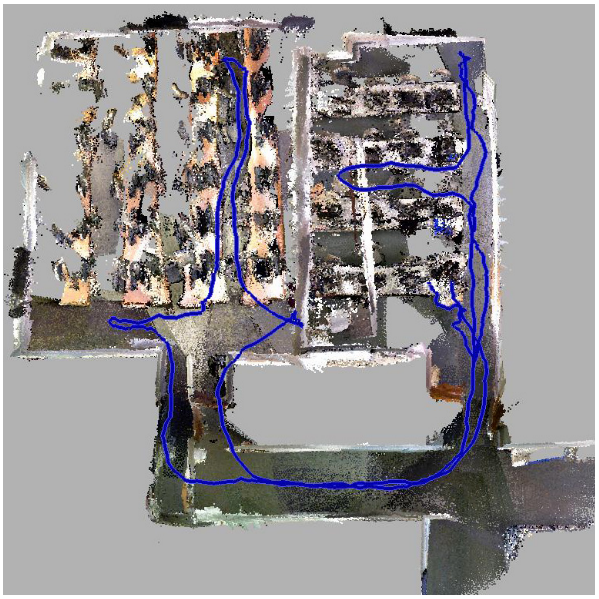


Figure 9: Example failure case.

3. Compute connected components by partitioning the image on silhouette boundaries.
4. Estimate normals for pixels using RANSAC on neighborhoods of radius 10cm within the same connected component.
5. Compute sets of coplanar pixels using hierarchical clustering.
6. Refine clusters with a RANSAC algorithm to reassign pixels to their largest compatible cluster.
7. Insert a proxy in P for each cluster. Assign the centroid, normal, and radius for the proxy based on PCA of the associated pixels.

Exracting features from images: The following sequence of operations is performed on the depth channel for each image $I[j]$ to extract features (see Section 4.1):

1. Construct a *planar surface feature* for each pixel in coplanar clusters of size ≥ 100.
2. Construct a *silhouette line feature* for each pixel marked silhouette, estimating the line direction with PCA on neighboring silhouette pixels.
3. Construct a *uniform surface feature* for every other pixel further than *min_spacing* from any previously created feature.

Computing errors: The following equations provide basic formulations used to define low-level error terms in Section 4.

We compute the coplanar misalignment E_{cp} of one planar entity A (represented by a position p_A

and a normal n_A) to another B (represented by p_B and n_B) by summing the squared distances from samples of points p_s ($s \in][1, smax]$) sampled from the center and on the boundary of a 1 meter radius disk around p_A to the plane of B ($smax$=5):

$$E_{cp}^{\rightarrow}(A,B) = \sum_{s=1}^{smax} ((p_A - p_B) \cdot n_B)^2$$

$$E_{cp}(A,B) = E_{cp}^{\rightarrow}(A,B) + E_{cp}^{\rightarrow}(B,A)$$

We compute the misalignment E_t of one rigid transformation T_j to another T_k in the neighborhood of a point c_j by summing the squared distances between points p_s ($s \in][1, smax]$) sampled uniformly on a 1 meter radius sphere when they are transformed by T_j versus T_k ($smax$=8):

$$E_t(T_j, T_k) = \sum_{s=1}^{smax} (T_j(p_s) - T_k(p_s))^2$$

These formulations provide error terms measured in squared distances between corresponding points (rather than differences of matrix elements, angles, viewpoints, etc.) and thus are more natural to combine with other error terms of the same form in our multi-objective optimization (as noted in [Pul99]).

Setting parameters: For all our experiments we have run the algorithm for 16 iterations with the window size creating 10 overlapping windows and increasing it linearly until it reaches n in the final iteration. We start with initial weights $w_L = 1000$, $w_S = 2000$, $w_C = 1000$, and final weights being $w_L = 1000$, $w_S = 1000$, $w_C = 2000$. We modify these weights by increasing them linearly until they reach maximum in 10th iteration, after which they stay fixed. This is done intuitively to allow for a couple of iterations when closest point alignment dominates.

This one set of parameters is used for every input (there is no tuning for specific examples). They were chosen intuitively without significant amounts of testing and refinement.

References

[AFDM08] Adrien Angeli, David Filliat, Stéphane Doncieux, and Jean-Arcady Meyer. Fast and incremental method for loop-closure detection using bags of visual words. *Robotics, IEEE Transactions on*, 24(5):1027–1037, 2008.

[AKJS12] Abhishek Anand, Hema S. Koppula, Thorsten Joachims, and Ashutosh Saxena. Contextually guided semantic labeling and search for 3d point clouds. *IJRR*, 2012.

[AMO] Sameer Agarwal, Keir Mierle, and Others. Ceres solver. `http://ceres-solver.org`.

[BM92] P. J. Besl and N. D. McKay. A method for registration of 3-D shapes. *IEEE Trans. PAMI*, 14(2):239–256, 1992.

[BS03] Adrien Bartoli and Peter Sturm. Constrained structure and motion from multiple uncalibrated views of a piecewise planar scene. *International Journal of Computer Vision*, 52(1):45–64, 2003.

[CBI13] Jiawen Chen, Dennis Bautembach, and Shahram Izadi. Scalable real-time volumetric surface reconstruction. *ACM Trans. Graph.*, 32(4):113:1–113:16, July 2013.

[CLH15] Kang Chen, Yu-Kun Lai, and Shi-Min Hu. 3d indoor scene modeling from rgb-d data: a survey. 1(4):267278, December 2015.

[CLJM14] Zetao Chen, Obadiah Lam, Adam Jacobson, and Michael Milford. Convolutional neural network-based place recognition. *arXiv preprint arXiv:1411.1509*, 2014.

[CZK15] Sungjoon Choi, Qian-Yi Zhou, and Vladlen Koltun. Robust reconstruction of indoor scenes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[Dav06] Timothy A. Davis. *Direct Methods for Sparse Linear Systems (Fundamentals of Algorithms 2)*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2006.

[DGFF12] Mingsong Dou, Li Guan, Jan-Michael Frahm, and Henry Fuchs. Exploring high-level plane primitives for indoor 3d reconstruction with a hand-held rgb-d camera. In *Asian Conference on Computer Vision*, pages 94–108, 2012.

[DNZ+16] Angela Dai, Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Christian Theobalt. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface re-integration. *arXiv preprint arXiv:1604.01093*, 2016.

[ENT05] C. Estrada, J. Neira, and J.D. Tardos. Hierarchical slam: Real-time accurate mapping of large environments. *Transactions on Robotics*, 21(4):588596, 2005.

[ERAB15] H. E. Elghor, D. Roussel, F. Ababsa, and E. H. Bouyakhf. Planes detection for robust localization and mapping in rgb-d slam systems. In *3D Vision (3DV), 2015 International Conference on*, pages 452–459, Oct 2015.

[FCSS09] Y. Furukawa, B. Curless, S.M. Seitz, and R. Szeliski. Manhattan-world stereo. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 0:1422–1429, 2009.

[FG14] Simon Fuhrmann and Michael Goesele. Floating scale surface reconstruction. *ACM Trans. Graph.*, 33(4):46:1–46:11, July 2014.

[FLD05] U. Frese, P. Larsson, and T. Duckett. A multilevel relaxation algorithm for simultaneous localisation and mapping. *IEEE Transactions on Robotics*, 21(2):112, 2005.

[GKSB10] G. Grisetti, R. Kmmerle, C. Stachniss, and W. Burgard. A tutorial on graph-based slam. *IEEE Intelligent Transportation Systems Magazine*, 2(4):31–43, 2010.

[HKH+10] 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 *International Symposium on Experimental Robotics (ISER)*, 2010.

[HWMD14] A. Handa, T. Whelan, J.B. McDonald, and A.J. Davison. A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. In *IEEE Intl. Conf. on Robotics and Automation, ICRA*, Hong Kong, China, May 2014.

[KLL+13] M. Keller, D. Lefloch, M. Lambers, S. Izadi, T. Weyrich, and A. Kolb. Real-time 3d reconstruction in dynamic scenes using point-based fusion. In *2013 International Conference on 3D Vision - 3DV*, pages 1 – 8, DOI:10.1109/3DV.2013.9, June 2013.

[KPR+15] O. Kahler, V. A. Prisacariu, C. Y. Ren, X. Sun, P. H. S Torr, and D. W. Murray. Very High Frame Rate Volumetric Integration of Depth Images on Mobile Device. *IEEE Transactions on Visualization and Computer Graphics (Proceedings International Symposium on Mixed and Augmented Reality 2015*, 22(11), 2015.

[LBF14] K. Lai, L. Bo, and D. Fox. Unsupervised feature learning for 3d scene labeling. In *IEEE International Conference on Robotics and Automation*, page 30503057, 2014.

[LSP08] Hao Li, Robert W. Sumner, and Mark Pauly. Global correspondence optimization for non-rigid registration of depth scans. *Computer Graphics Forum (Proc. SGP'08)*, 27(5), July 2008.

[LWC+11] Yangyan Li, Xiaokun Wu, Yiorgos Chrysanthou, Andrei Sharf, Daniel Cohen-Or, and Niloy J. Mitra. Globfit: Consistently fitting primitives by discovering global relations. *ACM Transactions on Graphics*, 30(4):52:1–52:12, 2011.

[MKSC16] L. Ma, C. Kerl, J. Stueckler, and D. Cremers. Cpa-slam: Consistent plane-model alignment for direct rgb-d slam. In *Int. Conf. on Robotics and Automation*, 2016.

[MMBM15] Aron Monszpart, Nicolas Mellado, Gabriel J. Brostow, and Niloy J. Mitra. Rapter: Rebuilding man-made scenes with regular arrangements of planes. *ACM Trans. Graph.*, 34(4):103:1–103:12, July 2015.

[MPM^{+}14] O. Mattausch, D. Panozzo, C. Mura, O. Sorkine-Hornung, and R. Pajarola. Object detection and classification from large-scale cluttered indoor scans. *Computer Graphics Forum*, 33(2):1121, 2014.

[NDI^{+}11] Richard A Newcombe, Andrew J Davison, Shahram Izadi, Pushmeet Kohli, Otmar Hillig̃es, Jamie Shotton, David Molyneaux, Steve Hodges, David Kim, and Ãndrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, pages 127–136. IEEE, 2011.

[NHS07] Viet Nguyen, Ahad Harati, and Roland Siegwart. A lightweight slam algorithm using orthogonal planes for indoor mobile robotics. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 658–663. IEEE, 2007.

[NRS15] T. Nguyen, G. Reitmayr, and D. Schmalstieg. Structural modeling from depth images. *IEEE Transactions on Visualization and Computer Graphics*, 21(11):1230–1240, Nov 2015.

[NZIS13] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger. Real-time 3d reconstruction at scale using voxel hashing. *ACM Transactions on Graphics (TOG)*, 2013.

[OVWK16] Sebastian Ochmann, Richard Vock, Raoul Wessel, and Reinhard Klein. Automatic reconstruction of parametric building models from indoor point clouds. *Computers & Graphics*, 54:94–103, 2016.

[PBVP] K. Pathak, A. Birk, N. Vaskevicius, and J. Poppinga. Fast registration based on noisy planes with unknown correspondences for 3-D mapping. *IEEE Trans. Robotics*, 26(3):424441, June.

[Pul99] Kari Pulli. Multiview registration for large data sets. In *Proceedings of the 2Nd International Conference on 3-D Digital Imaging and Modeling*, 3DIM'99, pages 160–168, Washington, DC, USA, 1999. IEEE Computer Society.

[RC11] R. Rusu and S. Cousins. 3d is here: Point cloud library (pcl). In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–4, 2011.

[RHHL02] Szymon Rusinkiewicz, Olaf Hall-Holt, and Marc Levoy. Real-time 3D model acquisition. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 21(3):438–446, July 2002.

[RL01] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the icp algorithm. In *3DIM*, pages 145–152. IEEE Computer Society, 2001.

[RS15] Adrian Ratter and Claude Sammut. Local map based graph slam with hierarchical loop closure and optimisation. 2015.

[SB08] Jorg Stuckler and Sven Behnke. Orthogonal wall correction for visual motion estimation. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 1–6, 2008.

[SDK09] R Schnabel, P Degener, and Reinhard Klein. Completion and reconstruction with primitive shapes. *Computer Graphics Forum*, 28(2):503–512, 2009.

[SEE^{+}12] 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.

[SHKF12] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *Proc. European Conf. on Comp. Vision*, 2012.

[SMGKD14] R.F. Salas-Moreno, B. Glocken, P.H.J. Kelly, and A.J. Davison. Dense planar slam. In *Mixed and*

Augmented Reality (ISMAR), 2014 IEEE International Symposium on, pages 157–164, Sept 2014.

[Sto16] Patrick Stotko. State of the art in real-time registration of rgb-d images. In *CESCG*, 2016.

[TF15] Yizhi Tang and Jieqing Feng. Hierarchical multiview rigid registration. *Comput. Graph. Forum*, 34(5):77–87, August 2015.

[TJRF13] Y. Taguchi, Yong-Dian Jian, S. Ramalingam, and Chen Feng. Point-plane slam for hand-held 3d sensors. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 5182–5189, May 2013.

[TL94] Greg Turk and Marc Levoy. Zippered polygon meshes from range images. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '94, pages 311–318, New York, NY, USA, 1994. ACM.

[TRC12] A. Trevor, J. Rogers III, and H. Christensen. Planar surface SLAM with 3D and 2D sensors. In *ICRA*, 2012.

[VLA15] Yannick Verdie, Florent Lafarge, and Pierre Alliez. LOD Generation for Urban Scenes. *ACM Transactions on Graphics*, 34(3):15, 2015.

[WACS12] Changchang Wu, Sameer Agarwal, Brian Curless, and Steven M. Seitz. Schematic surface reconstruction. In *Proc. Comp. Vision and Pattern Recognition*, 2012.

[WKF$^+$12] T. Whelan, M. Kaess, M.F. Fallon, H. Johannsson, J.J. Leonard, and J.B. McDonald. Kintinuous: Spatially extended KinectFusion. In *RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*, Sydney, Australia, Jul 2012.

[WKJ$^+$14] T. Whelan, M. Kaess, H. Johannsson, M.F. Fallon, J.J. Leonard, and J.B. McDonald. Real-time large scale dense RGB-D SLAM with volumetric fusion. *Intl. J. of Robotics Research, IJRR*, 2014.

[WLSM$^+$15] T. Whelan, S. Leutenegger, R. F. Salas-Moreno, B. Glocker, and A. J. Davison. ElasticFusion: Dense SLAM without a pose graph. In *Robotics: Science and Systems (RSS)*, Rome, Italy, July 2015.

[WS06] J. Weingarten and R. Siegwart. 3D SLAM using planar segments. In *IROS*, page 30623067, 2006.

[XOT13] Jianxiong Xiao, Andrew Owens, and Antonio Torralba. Sun3d: A database of big spaces reconstructed using sfm and object labels. *Computer Vision, IEEE International Conference on*, 0:1625–1632, 2013.

[ZK13] Qian-Yi Zhou and Vladlen Koltun. Dense scene reconstruction with points of interest. *ACM Transactions on Graphics*, 32(4), 2013.

[ZK14] Qian-Yi Zhou and Vladlen Koltun. Color map optimization for 3d reconstruction with consumer depth cameras. *SIGGRAPH Conf. Proc.*, 2014.

[ZK15] Qian-Yi Zhou and Vladlen Koltun. Depth camera tracking with contour cues. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[ZSN$^+$16] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, and Jianxiong Xiao. 3dmatch: Learning the matching of local 3d geometry in range scans. *arXiv preprint arXiv:1603.08182*, 2016.

[ZZP$^+$15] H. Zhou, D. Zou, L. Pei, R. Ying, P. Liu, and W. Yu. Structslam: Visual slam with building structure lines. *IEEE Transactions on Vehicular Technology*, 64(4):1364–1375, April 2015.