Joint Representation of Primitive and Non-Primitive Objects for 3D Vision

Christiane Sommer
Technical University of Munich
sommerc@in.tum.de

Daniel Cremers
Technical University of Munich
cremers@tum.de

Abstract

The use of structural information in 3D scanning is becoming more and more popular. However, most approaches exploit this structural information either in the form of geometric primitives (mostly planes) or known rigid bodies, but not both. We overcome this limitation and propose an object representation that combines primitive and non-primitive objects using one unified formulation that is based on signed distance fields. Object pose manifolds are introduced to represent the rigid movement of primitives and non-primitives in a natural way. We show that different components of volumetric scanning, such as global trajectory optimization or geometry completion and denoising, benefit from our formulation.

1. Introduction

Simultaneous localization and mapping (SLAM) has been a central topic in computer vision research for years. Depth sensors such as the Kinect allow for fast creation of dense 3D scene models, independent of lighting conditions. Low-cost depth sensors are primarily used in indoor environments, which are usually man-made and thus highly structured; many parts of a room can be described by planes (walls, floor, *etc*.) and rotationally symmetric or cylindric objects (trash cans, vases, *etc*.). Other rigid bodies appear repeatedly or can be reduced to one common class of shapes. Many approaches take advantage of this structure in indoor environments, *e.g*. by modeling planar patches explicitly or by estimating the 6D pose of known rigid bodies and integrating it into a SLAM routine. Ideally, one would like to exploit all types of structural information jointly and use it to aid tracking, ensure global map consistency, and improve reconstruction quality. The problem is that different types of objects have parameter spaces of different dimensionality and their parameters are determined differently (SVD for planes, ICP for rigid bodies, *etc*.). Simply treating planes and other geometric primitives as 6D rigid bodies will lead to degeneracies in the pose estimation.

In this work, we propose an object representation that combines geometric primitives, *i.e*. simple objects such as planes that can be described in closed form and usually have a lot of symmetries, and rigid bodies into one formulation. We decompose the object parameters into a shape and a pose part and define a signed distance field (SDF) as a function of these. This formulation can be used for joint object tracking, global trajectory optimization, and geometry completion and refinement. The object poses are manifold elements, which we explicitly take into account in the optimization to avoid degeneracies. When we use our model for tracking objects, the SDF-based representation provides an easy way to model sensor noise appropriately in the pose estimation process. Estimated object poses are used for "object bundle adjustment" to get a globally consistent camera trajectory. Finally, we propose ways to complete and refine scanning geometry. Examples are shown in Figure 1.

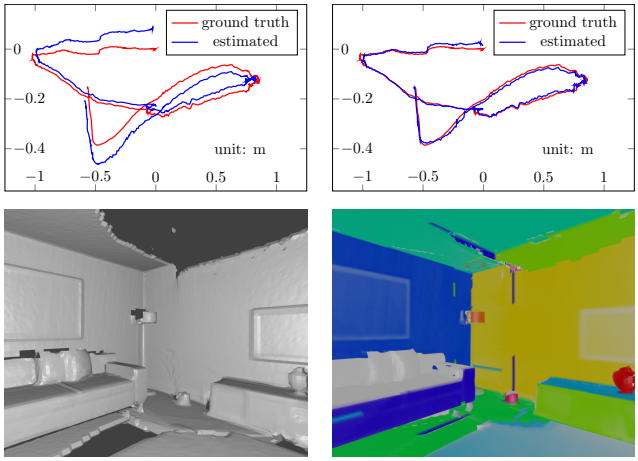

Figure 1. Applications of our object representation: we use object poses for global trajectory optimization (top right). After scanning, we fill in missing data in the reconstruction and denoise the resulting mesh using the objects (bottom right). The proposed structural knowledge not only improves the tracking precision, but it also provides a more complete geometry, most of which is grouped into known objects and geometric primitives such as planes and cylinders (colored). Left column is the reference implementation without object information (data: ICL-NUIM *lr2* [14]).

1.1. Related work

The foundation of 3D scanning with depth sensors was laid by Curless and Levoy [5], who introduced SDF voxel grids, and Rusinkiewicz *et al*. [28]. The KinectFusion system by Newcombe *et al*. [23] stores geometry as an SDF and tracks new frames against rendered depth maps using ICP. Bylow *et al*. [2] perform tracking directly against the SDF volume. Voxel hashing [25] and octrees [35] improve memory usage for these approaches to facilitate large scale volumetric scanning.

There is a lot of work on planar SLAM, *e.g*. [9, 22, 37, 38]. All of these use RGB-D input. The dense planar SLAM system for depth images by Salas-Moreno *et al*. [31] uses a surfel representation of the scene and compresses data by jointly storing information for surfels belonging to the same plane. Planes are not used to enforce global consistency, and no other structural information is used for the non-planar surfels. Dzitsiuk *et al*. [10] use plane priors to complete and denoise the final reconstruction, but not for camera tracking. For RGB-D scans, Huang *et al*. [16] denoise and simplify meshes using planes.

In the field of semantic scanning and reconstruction, the SLAM++ system by Salas-Moreno *et al*. [32] achieves high data compression in repetitive scenes. Poses of objects in a database are refined using ICP after detection and then used for global graph optimization. The semantic bundle adjustment by Fioraio *et al*. [13] runs on RGB sequences. They use 2D image features in addition to the 6D semantic features, which makes their approach more generally applicable than the SLAM++ system. We are not aware of any work combining semantic and primitive-based scanning consistently.

Zhang *et al*. [39] use planes and rigid bodies to describe a scene. The rigid bodies are reconstructed on-the-fly, and incoming data is labeled as either plane or object. However, plane and object pose parameters are not used for tracking, and no geometric primitives other than planes are part of their model description.

The *g2o* framework by Kümmerle *et al*. [18] is a generic graph optimization system, which allows for different types of graph vertices, *i.e*. objects. Hertzberg *et al*. [15] have presented a sensor fusion framework based on manifold formulations. The object pose part of our formulation is similar, but we additionally have an explicit SDF-based geometry description which allows for applications on the reconstruction side of a SLAM algorithm.

1.2. Contributions

We propose to unify the description of geometric primitives and non-primitive rigid objects for 3D tracking and mapping tasks. The main contributions of our work are:

- We present a new formulation based on SDFs that allows us to represent all types of objects (including, but not limited to planes and 6D rigid bodies) jointly, irrespective of their dimensionality.
- We define object pose manifolds and provide a thorough analysis of the most relevant ones.
- We show how our joint object representation can be integrated into different stages of a volumetric depth SLAM system to improve tracking and mapping and reduce the memory footprint.

2. Object representation

To represent a rigid body in space, we need two things: its pose $(R, \mathbf{t})$, *i.e*. its rotation and translation w.r.t. a reference coordinate system, and its shape. The shape can be described explicitly – *e.g*. by a triangulation – or implicitly – as an occupancy grid or a signed distance field (SDF).

We generalize this concept of pose and shape parameters to arbitrary object types: each object is described by a set of pose parameters $\mathbf{p} = (p^1, ..., p^m)$ that change when the object moves w.r.t. the camera frame and a set of shape parameters $\mathbf{s} = (s^1, ..., s^n)$ that are fixed for one particular instance of an object. The pose parameters uniquely represent elements p of the object pose manifold $\mathcal{M}$. The space of shape parameters $\mathcal{S}$ does not need to admit a manifold structure. Knowing $p \in \mathcal{M}$, represented by the parameter vector $\mathbf{p}$, and $\mathbf{s} \in \mathcal{S}$, we define a signed distance field

$$\psi_o : \mathbb{R}^3 \to \mathbb{R} \tag{1}$$

that returns for any point $\mathbf{x} \in \mathbb{R}^3$ the signed Euclidean distance to the surface of the object $o := (p, \mathbf{s})$ [26, Chapter I.2]. For geometric primitives, ψ_o can be stated in closed form. We define $\psi_o(\mathbf{x}) > 0$ for $\mathbf{x}$ inside an object, and $\psi_o(\mathbf{x}) < 0$ for a point $\mathbf{x}$ outside the object.

2.1. Object pose manifolds

By optimizing directly on the manifold, we prevent parameterization-induced singularities in the optimization. For the general definition of a manifold, we refer to any textbook on differential geometry, *e.g*. [27]. To implement manifold elements as simple *Vector* objects, we require:

1. An injective map $\mathcal{P} : \mathcal{M} \to \mathbb{R}^m$ that assigns each manifold element p its parameter vector $\mathbf{p}$. Once $\mathcal{P}$ is defined, p is identified with $\mathbf{p}$ to not clutter notation.

In order to perform on-manifold optimization in a consistent way, we need (see [1]):

2. For every point $p \in \mathcal{M}$ a retraction $r_p : T_p\mathcal{M} \to \mathcal{M}$ that maps tangent vectors in $T_p\mathcal{M}$ back to $\mathcal{M}$.
3. An on-manifold gradient $\nabla_{\mathcal{M}}$.

For an object pose manifold, we require in particular:

4. A (semi-)metric $d : \mathcal{M} \times \mathcal{M} \to \mathbb{R}_{\geq 0}$ for which d^2 is a weighted sum of the squared Euclidean norms of a translational and an angular distance vector:

$$d(p_1, p_2)^2 = \omega_t \|\mathbf{r}_t(p_1, p_2)\|^2 + \omega_r \|\mathbf{r}_r(p_1, p_2)\|^2. \tag{2}$$

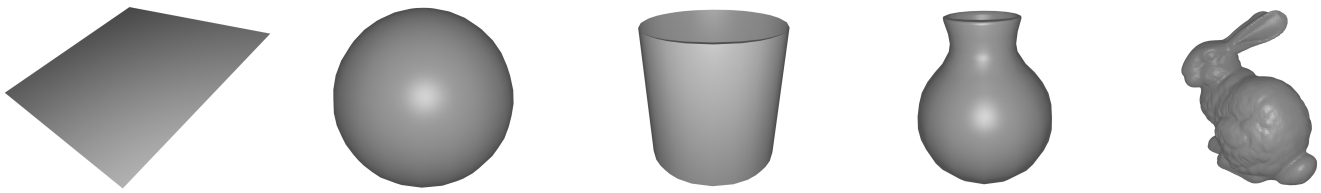

Figure 2. The five object types that we analyze: plane and sphere poses have three degrees of freedom (DoF), cylinder poses four. The vase is an example of a solid of revolution (5 DoF), and the bunny is an example of a rigid body. Its pose has the maximum of 6 DoF.

This is important for optimization methods that require (weighted) sums of squared residuals, such as Gauss-Newton or Levenberg-Marquardt.

5. An $SE(3)$ Lie group action $\phi : SE(3) \times \mathcal{M} \to \mathcal{M}$, $(T, p) \mapsto T \cdot p$. It describes how p, seen from a camera, changes if object or camera are moved rigidly. We define $\cdot$ implicitly through

$$\psi_{(T\cdot p,\mathbf{s})}(T \cdot \mathbf{x}) = \psi_{(p,\mathbf{s})}(\mathbf{x}) \,, \tag{3}$$

i.e. $\psi_o(\mathbf{x})$ must not change if $\mathbf{x}$ and p move rigidly in the same way.

4. and 5. distinguish our manifold specifications from those of Hertzberg *et al.* [15]: our (semi-)metric has an actual geometric interpretation, and the $SE(3)$ action only makes sense for manifolds whose elements represent object poses.

2.2. Examples of object spaces

In the following, we briefly describe the object pose manifolds of five common object types. $\mathcal{M}$ (including $\mathbf{r}_t$ and $\mathbf{r}_r$) as well as shape spaces $\mathcal{S}$ and their SDFs are summarized in Figure 2 and Table 1. A more in-depth analysis containing details on the used retractions, metrics and semi-metrics can be found in the Appendix.

Oriented planes are fully described by a normal vector $\mathbf{n}$ and a distance d. They do not have any shape parameter.

To characterize a **sphere**, we need its center point $\mathbf{c}$ and the radius r. $\mathbf{c}$ changes when the coordinate system is moved rigidly, while the radius r is a shape parameter and thus remains fixed.

Cylinders are described by the cylinder axis (pose) and the radius r (shape). The axis is a one-dimensional affine subspace of $\mathbb{R}^3$. Thus, $\mathcal{M} = \mathrm{Graff}_1(\mathbb{R}^3)$, the Grassmannian manifold of affine 1D subspaces of $\mathbb{R}^3$. Following [17], we parameterize $\mathrm{Graff}_1(\mathbb{R}^3)$ by a direction vector $\mathbf{n} \in \mathbb{S}^2 \subset \mathbb{R}^3$ and an offset $\mathbf{d} \perp \mathbf{n}$ in $\mathbb{R}^3$.

The pose of a **solid of revolution**, *i.e.* a rotationally symmetric rigid body, is a position $\mathbf{d}$ in 3D space together with an axis direction $\mathbf{n}$. Its shape space is the space of all plane curves, but can in practice be limited to a finite number of parameters that encode a certain class of shapes, *e.g.* by performing PCA on a set of curves. We can deduce the 3D SDF ψ_o of a solid of revolution from the 2D SDF $\psi^{2D}_{\mathbf{s}}$ of the corresponding plane curve.

All mentioned objects are rigid bodies, with symmetries reducing the dimensionalities of their pose manifolds. In this work, we use the term **rigid body** exclusively for objects, the pose manifold of which is the space $SE(3)$ of rigid body motions. This most general type of object has no continuous symmetries. As for the solid of revolution, its shape can be parameterized by some low-dimensional representation of a class of shapes, using *e.g.* PCA to approximate the SDFs of a set of sample shapes [4, 11, 19]. The rigid body pose T is an element of the Lie group $SE(3)$. A useful local parameterization is given by the Lie algebra $\mathfrak{se}(3)$. We compute the rigid body SDF ψ_o from the SDF $\psi^0_{\mathbf{s}}$ of a given object instance.

2.3. Collections of objects

A scene that consists of N objects is the union of all these objects. We define a scene distance function (DF) Ψ as the pointwise minimum of all absolute values of object SDFs:

$$\begin{aligned} \Psi &: \prod_{j=1}^{N} (\mathcal{M}_j \times \mathcal{S}_j) \times \mathbb{R}^3 \to \mathbb{R}_{\geq 0} \,, \\ \Psi(o_1, ..., o_N, \mathbf{x}) &:= \min_{j=1,...,N} |\psi_{o_j}(\mathbf{x})| \,. \end{aligned} \tag{4}$$

The argmin provides an intrinsic labeling of $\mathbb{R}^3$: let

$$j_0(\mathbf{x}) = \operatorname*{argmin}_{j=1,...,N} |\psi_{o_j}(\mathbf{x})| \,, \tag{5}$$

then object j_0 is the object closest to point $\mathbf{x}$. Note that even though Ψ is not differentiable in $\Psi(\cdot, \mathbf{x}) = 0$, Ψ^2 is and thus, $\nabla\Psi^2(\cdot, \mathbf{x})$ is well-defined. An illustrative example for a 2D scene is shown in Figure 3. The possibility of representing the scene with one combined DF allows us to jointly estimate all object poses at once.

The following sections show exemplary applications of our scene DF representation for object pose tracking, global optimization, and geometry extraction of volumetric scanning algorithms.

3. Frame-to-frame joint object tracking

Given a set of object parameters in one depth image, we want to track the objects, *i.e.* estimate their poses in the subsequent one. This avoids the problem of data association of

Object	$\mathcal{M}$	$\dim \mathcal{M}$	$\mathbf{r}_t(p_1, p_2)$	$\mathbf{r}_r(p_1, p_2)$	$\mathcal{S}$	$\psi_{(p,\mathbf{s})}(\mathbf{x})$
Plane	$\mathbb{S}^2 \times \mathbb{R}$	3	$d_1 - d_2$	$\mathbf{n}_1 - \mathbf{n}_2$	$\emptyset$	$\mathbf{n}^\top \mathbf{x} + d$
Sphere	$\mathbb{R}^3$	3	$\mathbf{c}_1 - \mathbf{c}_2$	0	$\mathbb{R}_{\geq 0}$	$r - \|\mathbf{x} - \mathbf{c}\|$
Cylinder	$\mathrm{Graff}_1(\mathbb{R}^3)$	4	$\mathbf{m}\mathbf{m}^\top(\mathbf{d}_1 - \mathbf{d}_2)$ $\mathbf{m} = \frac{\mathbf{n}_1 \times \mathbf{n}_2}{\|\mathbf{n}_1 \times \mathbf{n}_2\|}$	$\mathbf{n}_1 - \mathrm{sgn}(\mathbf{n}_1^\top \mathbf{n}_2)\mathbf{n}_2$	$\mathbb{R}_{\geq 0}$	$r - \sqrt{\|\mathbf{x} - \mathbf{d}\|^2 - (\mathbf{n}^\top \mathbf{x})^2}$
Solid of rev.	$\mathbb{S}^2 \times \mathbb{R}^3$	5	$\mathbf{d}_1 - \mathbf{d}_2$	$\mathbf{n}_1 - \mathbf{n}_2$	-	$\mathrm{interpolate}(\psi_{\mathbf{s}}^{2D}, (\rho, h))$ $h(\mathbf{x}) = \mathbf{n}^\top(\mathbf{x} - \mathbf{d})$ $\rho(\mathbf{x}) = \sqrt{\|\mathbf{x} - \mathbf{d}\|^2 - h(\mathbf{x})^2}$
Rigid body	$SE(3)$	6	$\log({T_1}^{-1} T_2)_{1:3}$	$\log({T_1}^{-1} T_2)_{4:6}$	-	$\mathrm{interpolate}(\psi_{\mathbf{s}}^0, T^{-1} \cdot \mathbf{x})$

Table 1. Object pose manifolds with metric residuals, shape spaces and SDFs for five common object types. For solids of revolution and rigid bodies, no concrete shape space is defined, since there are multiple ways to define one. The cylinder residual $\mathbf{r}_t$ is $\mathbf{d}_1 - \mathbf{d}_2$ if $\mathbf{n}_1 \parallel \mathbf{n}_2$.

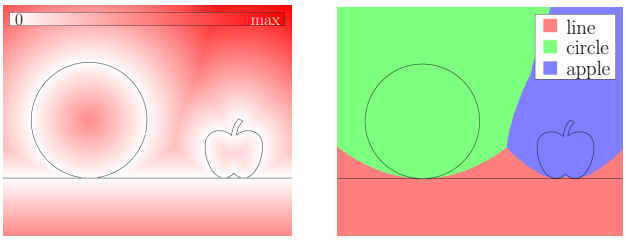

Figure 3. 2D example of a scene consisting of a set of different types of objects: shown are min (left) and argmin (right) of $|\psi_j(\mathbf{x})|$, *i.e.* scene distance function and intrinsic labeling. Circle and line SDF can be written in closed form, while the apple is a rigid body lacking a closed form SDF. We store such SDFs of non-primitives using pixel (2D) or voxel (3D) grids.

detection-based approaches. The manifold formulation ensures non-degenerate solutions. We assume a rigid setting, *i.e.* shape parameters are known and fixed during tracking.

3.1. Probabilistic object pose estimation

For the given depth data, we assume two things: the noise on the data points is Gaussian, and it is symmetric w.r.t. the z-axis. This is justified by analyses on Kinect noise characteristics, see *e.g.* [24]. z-symmetric noise implies a diagonal covariance matrix Σ with $\sigma_x = \sigma_y =: \sigma_{xy}$. The probability for a data point $\mathbf{x}$ if the true (closest) point on the scene surface is $\mathbf{x}_0$ is

$$p(\mathbf{x}|\mathbf{x}_0) = \frac{1}{\sqrt{(2\pi)^3 \det(\Sigma)}} \exp\left(-\frac{1}{2} \|\mathbf{x} - \mathbf{x}_0\|_\Sigma^2\right) . \quad (6)$$

We can rewrite $\mathbf{x} - \mathbf{x}_0$ in terms of the scene DF Ψ, exploiting the distance field property (see Appendix):

$$\mathbf{x} - \mathbf{x}_0 = \nabla \Psi^2(\cdot, \mathbf{x}) . \quad (7)$$

Using $\|\nabla \Psi(\cdot, \mathbf{x})\| = 1 \; \forall \, \mathbf{x}$, the Malahanobis distance in Eq. (6) becomes (see Appendix)

$$\|\mathbf{x} - \mathbf{x}_0\|_\Sigma^2 = w(\mathbf{x}, \nabla\Psi(\cdot, \mathbf{x}))\Psi(\cdot, \mathbf{x})^2 \quad \text{with}$$
$$w(\mathbf{x}, \nabla\Psi(\cdot, \mathbf{x})) = \frac{1}{\sigma_{xy}^2} + (\partial_z \Psi(\cdot, \mathbf{x}))^2 \left(\frac{1}{\sigma_z^2} - \frac{1}{\sigma_{xy}^2}\right) . \quad (8)$$

To obtain $p(\mathbf{x}|S_k)$, where S_k is shorthand notation for all observed objects in the scene in frame k, we have to integrate $p(\mathbf{x}|\bar{\mathbf{x}})$ over all $\bar{\mathbf{x}}$ that lie on the surface of S_k. Since $p(\mathbf{x}|\bar{\mathbf{x}})$ cannot be expressed as a function of Ψ, we approximate $p(\mathbf{x}|S_k) \approx p(\mathbf{x}|\mathbf{x}_0)$. Using Bayes' theorem and assuming a uniform prior on the scene geometry and independence of data points yields

$$p(S_k|\{\mathbf{x}_i\}_i) \propto p(\{\mathbf{x}_i\}_i|S_k) \approx \prod_i p(\mathbf{x}_i|\mathbf{x}_{0,i}) , \quad (9)$$

see the Appendix for a derivation. Thus, we can define a cost function as the negative log likelihood

$$E_{\text{track}}(S_k) := -\log p(S_k|\{\mathbf{x}_i\}_i)$$
$$\approx \frac{1}{2} \sum_i w(\mathbf{x}_i, \nabla\Psi(S_k, \mathbf{x}_i))\Psi(S_k, \mathbf{x}_i)^2 + \text{const} . \quad (10)$$

3.2. Optimization

To minimize the objective from Eq. (10) (and thus maximize the likelihood for a scene configuration S_k), we interpret it as an iteratively re-weighted least squares problem by fixing the weights $w(\mathbf{x}_i, \nabla\Psi(S_k, \mathbf{x}_i))$ during each iteration and updating them before the next one.

The domain of E_{track} is the cross-product manifold $\prod_j \mathcal{M}_j$. We use on-manifold Gauss-Newton optimization as described in [1] to solve for the observed object poses $S_k = (\bar{p}_{k,0}, ..., \bar{p}_{k,N-1})$. The pointwise minimum in Ψ

leads to a block-diagonal structure of the Hessian; and, thus, the problem reduces to N subproblems of low dimensionality in each iteration, while still optimizing for all object poses jointly. Just like Gauss-Newton in Euclidean space, this procedure needs good initialization to converge. We use the scene configuration S_{k-1} as the initial estimate for S_k. To initialize S_0 or detect previously unseen objects, any (primitive) object detection method for point cloud data can be used.

4. Consistency of object and camera poses

If we assume that the scene is static, we expect consistency between frames: the observed pose $\bar{p}_{kj}$ of object j in frame k should be the object pose in the coordinate system of camera $C_k \in SE(3)$: $\bar{p}_{kj} = C_k^{-1} \cdot p_j$.

4.1. Enforcing consistency

This consistency constraint can be imposed by directly minimizing a joint energy over all frames. However, this would remove the possibility of running the optimization online, while new frames come in.

We thus follow another line of reasoning: having computed $\bar{p}_{kj}$ independently for the first K frames, we achieve consistency by projection of $\{\bar{p}_{kj}\}_{k,j}$ to the space of consistent object poses, *i.e.* those that can be written as $C_k^{-1} p_j$. The projection is updated each time a new frame is tracked. We minimize the squared on-manifold distance d^2 between $\bar{p}_{kj}$ and $C_k^{-1} p_j$ for all $k > 0$ and j jointly:

$$E_{\text{pr}}(\{C_k\}_{k>0}, \{p_j\}_j) := \sum_{k,j} w_{kj} d_j(\bar{p}_{kj}, C_k^{-1} p_j)^2 \,, \quad (11)$$

with weights w_{kj} that take into account size and visibility of the object. We propose a choice of weights in the results section.

4.2. Relation to bundle adjustment

This projection energy has the same structure as the standard multi-view bundle adjustment (BA) reprojection error. Thus, we can apply algorithms developed for classical BA and combine them with optimization methods for manifolds to do the projection step.

Specifically, we adapt the algorithm proposed by Engels *et al.* [12] to work with manifold input of different dimensionalities. It performs Levenberg-Marquardt minimization on the BA energy and uses the Schur complement to reduce complexity from $\mathcal{O}((K+N)^3)$ for a naive implementation to $\mathcal{O}(K^2 N)$. We swap K and N in our implementation to get a complexity of $\mathcal{O}(KN^2)$, since N is typically smaller than K in our case. We use robust Cauchy weights as suggested in [12] and represent the camera poses C_k by twist coordinates in $\mathfrak{se}(3)$.

Figure 4. Limitations of primitive-based models: while the 6D camera motion is clearly constrained in both scenes, a model that only detects planes will not be able to determine the x-component of the camera motion, since none of the plane normals have non-zero x-component.

4.3. Regularization

Similarly to the BA problem, where at least five non-coplanar points are needed for a unique solution, there will not always be a unique minimizer for Eq. (11). In particular, since planes and cylinders extend to infinity in our model, *e.g.* edges of cubes will not necessarily provide enough information to constrain the camera pose. We give an example of this in Figure 4.

To solve this problem, we regularize the object bundle adjustment using information from a global frame-to-model fitting. We use a coarse voxel grid to save an estimate of the scene SDF volumetrically, following Bylow *et al.* [2]. When a new frame comes in, in addition to the object tracking, the camera pose C_k^0 that best aligns the SDF voxel grid with the depth data is estimated. We constrain the object BA by adding a regularization term

$$E_{\text{reg}}(\{C_k\}_k) = \sum_k \bar{w}_k d_{SE(3)}(C_k^0, C_k)^2 \quad (12)$$

to the energy E_{pr} in Eq. (11). This term ensures that the camera pose C_k stays close to that estimated from the low resolution SDF grid. The weights $\bar{w}_k$ are specified in the results section. This does not change the complexity of our BA implementation.

Note that the resolution of the voxel grid can be low compared to [2], since we mainly use it to regularize our method. We have closed form SDFs for the geometric primitives. Thus, assuming most parts of the scene are decribed by the scene DF, fine-scale geometry need not be stored explicitly.

It is also possible to include a frame-to-frame regularizer and solve the problem in a pose graph optimization manner, *e.g.* using the *g2o* framework [18].

5. Geometry completion and denoising

When scanning a room, certain areas in the room might not be observed at all or hardly be observed from any camera position. This can lead to missing or very noisy data in

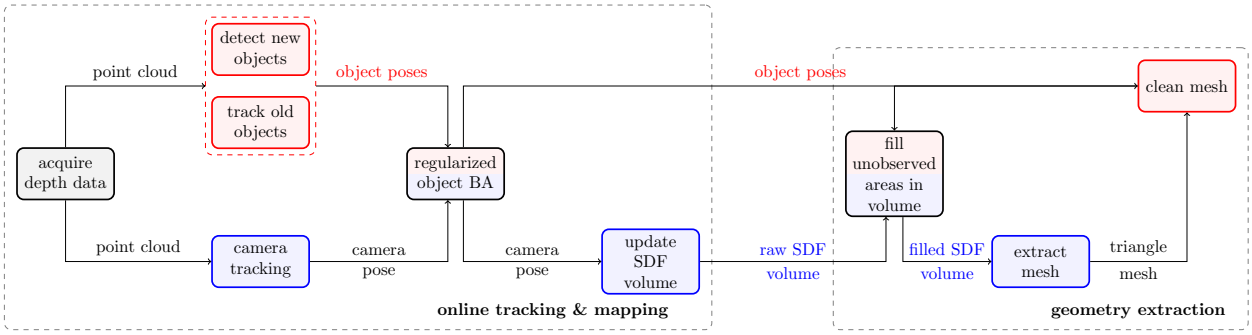

Figure 5. Overview of a generic volumetric scanning workflow (blue) augmented with structural information about arbitrary object types (red): object pose and shape information can be used at several stages of the scanning process to attain a globally consistent model.

the resulting mesh. The object information encoded in the scene DF can be used to resolve this issue.

5.1. SDF completion

We fill in the missing data with our prior knowledge: we have a volumetric SDF grid as in [2] with distances d and weights w for each voxel. For each voxel $\mathbf{v}$ with weight zero, we replace the distance with the SDF value of its closest object $j_0(\mathbf{v})$. This way, we obtain an SDF volume with data everywhere. Note that this SDF can differ from the true one at the voxel positions with $w = 0$.

5.2. Mesh denoising

We extract a mesh from the completed SDF using Marching Cubes [21]. Due to noise in depth data and estimated camera poses, this mesh will be noisy. We use the object information in the scene DF to smoothen it: in a scene that is perfectly described by a set of known objects, the DF property from Eq. (7) allows us to project each mesh vertex $\mathbf{x}$ to its corresponding point on the surface $\mathrm{proj}(\mathbf{x})$:

$$\mathrm{proj}(\mathbf{x}) = \mathbf{x} - \nabla\Psi^2(\cdot, \mathbf{x})\,. \tag{13}$$

In a real world scenario, there will always be parts of the scene that are not described by known objects. We thus define a threshold t above which we do not project points:

$$\mathrm{proj}_t(\mathbf{x}) = \begin{cases} \mathbf{x} - \nabla\Psi^2(\cdot, \mathbf{x}) & \text{if } \Psi(\cdot, \mathbf{x}) \leq t \\ \mathbf{x} & \text{else} \end{cases} \tag{14}$$

This is equivalent to truncating Ψ at t.

5.3. Mesh subdivision

Optimally, a large part of our scene is described by primitive or non-primitive known objects, and the resolution of the regularization voxel grid can be quite low. This saves memory. However, at the same time, the geometry of smaller objects becomes inaccurate in the resulting mesh. If these smaller objects are (partly) described by known objects or primitives (*e.g.*, a Coke can by a cylinder), we can restore finer geometry by mesh subdivision [20]: we divide each triangle in the mesh into four smaller triangles by connecting the midpoints of the three edges. These midpoints are then re-projected to the scene surface using Eq. (14). Since we refine geometry only after mesh extraction, the memory required for the voxel grid does not increase.

6. Experiments and results

6.1. Implementation

We implemented two abstract base classes, *sdf_unit* and *manifold* in C++, from which we derived the explicit examples of object types and pose manifolds. The scene is represented by a vector of *sdf_unit*s.

For the regularization grid, we implemented the method of Bylow *et al.* [2]. The same implementation is used as reference implementation for pure volumetric scanning.

Primitives are detected with the method by Schnabel *et al.* [33]. Rigid bodies and solids of revolution are detected using FPFH features [29]. We use Point Cloud Library [30] for 3D point processing (*e.g.* normal computation for object detection). New objects are detected every 30th frame. Object BA is done with a window size of five frames while scanning, and one global BA using all information is performed after scanning. To not lose tracking of objects, poses of objects that are not visible are updated using the estimated camera pose.

6.2. Parameters and setting

For speed reasons, we evaluate our approach on depth images with QVGA resolution, 320×240. Since primitives only need very few data points to be fully characterized, we believe that the reduced resolution does not deteriorate results much. We take the depth channel of RGB-D data and downsample it by a factor of two in each dimension.

Usually, not every part of the scene belongs to a geometric primitive or known object, so we need some outlier handling in our tracking. The most widely used technique to handle outliers in SDF-based fitting is to truncate the SDF:

$\Psi_T = \max(\min(\Psi, T), -T)$ for some $T \in \mathbb{R}$. We choose a data-dependent truncation value $T = 2\sigma_z(\mathbf{x})$. Note that the T here and t from Eq. (14) are not necessarily related.

σ_z and σ_{xy} are chosen depending on the sensor. For Kinect data, we use the model proposed by Nguyen *et al*. [24] and simplify it using polynomials. For the synthetic ICL-NUIM data, we use the noise model described in [14]. Details can be found in the Appendix.

As weights in the object BA, we choose $w_{kj} = N_{kj}^2 / \sum_i \psi_{o_j}(\mathbf{x}_i^{(k)})^2$, where N_{kj} is the number of points belonging to object j in frame k, and the sum is over all these points. This choice takes into account both the visibility of the object and the distance of the points from the object surface. More elaborate choices, *e.g*. related to the Fisher information, have not proven advantageous but have a higher computational cost. The regularization weights $\bar{w}_k$ are chosen accordingly as the ratio of the number of points N_k in frame k and their mean squared distance from the low resolution scene surface.

6.3. Synthetic scenes

We run our implementation on two *living room* sequences of the ICL-NUIM benchmark [14]. This is a good example for a scene where the proposed approach is useful, since it contains different sorts of objects: floor, ceiling, walls and parts of the sofas are planar; the lamp shades are cylindrical; the vase is a solid of revolution; and the chairs are rigid bodies without continuous symmetries. One of the advantages of our approach compared to SLAM++ is that not all objects need to be known, so we assume that we only know the geometry of the vase but not of the chairs. To compare to other methods, we compute statistics of the *absolute trajectory error* (ATE) of the estimated camera poses as suggested by Sturm *et al*. [36]. We report the ATE for the *lr1* and *lr2* sequences compared to [31] and [3] in Table 2. In summary, we obtain results comparable to the reference approaches. Additionally, our model is able to recognize not only planar parts but also *e.g*. the cylindric structure of the lamp shades, which can help to refine geometry.

In Figure 6, we show that using object information, the ATE remains reasonably low even for coarse voxel grids: while the mean ATE for a voxel size of $(8\,\mathrm{cm})^3$ is 6.00 cm using volumetric SDF scanning only, it goes down to 2.71 cm when using object poses, which is more than twice as accurate. However, erroneous object detection can have a negative impact on tracking and thus lead to high ATEs. Better object detection might improve the object-assisted scanning, especially w.r.t. robustness.

A reconstruction of the living room with a completed and denoised geometry is given in Figure 1. Despite some wrong geometry predictions in unobserved areas similar to [10], overall the reconstruction improves when the object information is used.

Dataset	ICL-NUIM *lr1*			ICL-NUIM *lr2*		
Method	[31]	(a)	(b)	[3]	(a)	(b)
RMSE	**1.69**	2.32	2.14	3.3	4.89	**1.90**
Mean	**1.50**	1.99	1.73	-	4.45	1.77
Median	1.64	1.84	**1.28**	-	3.64	1.70
Std.	0.78	1.19	1.27	-	2.03	0.70
Min.	0.22	0.11	0.17	-	1.73	0.37
Max.	2.84	6.55	5.48	-	10.19	3.71

Table 2. ATE statistics [cm] for two *living room* sequences of the ICL-NUIM dataset [14]: [31] is dense planar SLAM method using surfels; [3] is robust reconstruction by Choi *et al*. (result taken from [6]). (a) denotes volumetric scanning with a linear voxel size of 4 cm; (b) is the same augmented by object pose information. Despite the big voxel size and QVGA input, we obtain trajectory errors comparable to other depth-only methods. In particular, volumetric scanning benefits greatly from the structural information.

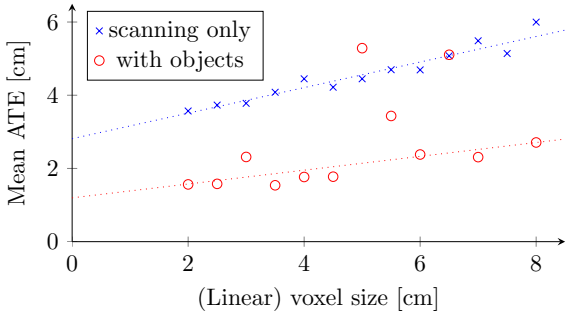

Figure 6. Mean absolute trajectory errors for a naive implementation of SDF-based tracking and mapping (blue) [2] and the same model augmented with the (primitive) object tracking and bundle adjustment (red): except for some outliers (*e.g*. at 5 cm) which are due to wrong object detections, the use of structural information almost halves the mean ATE, *i.e*. improves the trajectory a lot.

6.4. Real Kinect data

We use the *Kinect bunny handheld* sequence from the 3D Printed dataset [34] as an example for real depth sensor data, because the dataset provides a ground truth model of the bunny. Except for the bunny, it has two planes that can be used for tracking. To model Kinect sensor noise, we employ the model from [24]. The natural method to compare to is the SDF2SDF method from the dataset paper, which is also a depth-only method. Following their work, we report translational and rotational parts of the *absolute pose error* (APE) instead of the ATE. The APE is similar to the ATE, but uses a different method to align the two trajectories. Results are shown in Table 3. In summary, we improve on the baseline despite a much coarser resolution.

We also demonstrate geometry refinement using the bunny as an example. Figure 7 shows that after mesh de-

Method	SDF2SDF		Scan only		With objects	
Metric	trans	rot	trans	rot	trans	rot
RMSE	4.10	-	4.93	6.31	**1.69**	1.92
Mean	3.76	4.58	4.53	5.79	**1.54**	**1.75**
Min.	0.12	0.00	0.23	0.34	0.12	0.11
Max.	7.81	7.79	8.98	12.68	3.52	6.30

Table 3. APE statistics [cm/deg] for the *Kinect bunny handheld* sequence [34]: even with a (linear) voxel size eight times larger than the reference method (2.0 cm *vs*. 2.5 mm) and QVGA input, pose information for three objects (2 planes, 1 rigid body) helps to reduce the absolute pose error significantly.

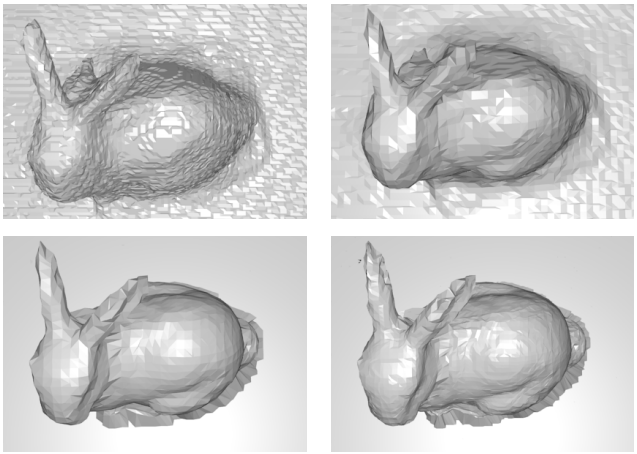

Figure 7. Geometry refinement for the *Kinect bunny handheld* [34]: pure volumetric scanning with voxel length 2.5 mm (top left) blurs details such as the tail of the bunny and introduces artifacts (between the ears). Even for a coarser resolution (voxel length 5.0 mm), these problems are resolved when we integrate object knowledge to scan (top right), denoise geometry (bottom left) and refine the mesh by subdivision (bottom right). Thus, we are able to increase scan quality while reducing scan memory.

noising and subdivision, coarse scanning using object pose and shape information is superior to fine-grained scanning without object knowledge: it needs less memory, but still produces scans with a higher level of detail. More evaluation of this sequence is provided in the Appendix.

6.5. Runtime and memory

We used a standard desktop PC with an Intel Xeon CPU at 3.5 GHz. For the sequences used in the results, joint object tracking takes on average 55 ms per frame, *i.e*. 18 fps. The bundle adjustment part, which can run in parallel to the tracking, is at a similar rate of 17 fps for 880 frames. Object detection runs considerably slower, on the order of 1.0 s.

To get an idea about memory requirements, we give an example in Table 4. Since scan volume resolution can be lower for scanning with structural information, less memory is needed. This is also true for memory-efficient methods like voxel hashing [25], because the low resolution scan volume can be stored more efficiently as well. The compression rate will be smaller in this case.

Volumetric scanning only	
2×256^3 voxels (double)	**256 MB**
Scanning with objects	
coarse SDF: 2×85^3 voxels (double)	9.37 MB
12 planes, 880 measurements each	1.46 MB
2 cylinders, 880 measurements each	429 kB
1 solid of revolution, 880 measurements	153 kB
SDF for solid of revolution, 98×43 pixels	32.9 kB
total	**11.4 MB**

Table 4. For the ICL-NUIM *lr2* sequence [14], we compare memory requirements of volumetric scanning with $(2\,\mathrm{cm})^3$ voxels to coarse scanning with $(6\,\mathrm{cm})^3$ voxels augmented by 15 object poses. Both have similar ATEs. For a volume of $(5.12\,\mathrm{m})^3$, the data compression rate is 22, *i.e*. between that of dense planar SLAM and SLAM++ [31, 32].

7. Conclusion

We have introduced a novel framework for representing a 3D world containing different types of objects which can facilitate object-assisted 3D scanning. This was achieved by decomposing the parameter space of an object into an object pose manifold and a shape space and defining all object types consistently by signed distance fields. Only pose parameters change when the object or coordinate system moves rigidly. The manifold formulation avoids degeneracies in the pose estimation. Our representation is integrated into an SDF-based tracking and mapping approach, where it helps to improve the global trajectory. Moreover, the structural information is used to denoise and refine the reconstructed geometry while keeping memory requirements low.

We believe that the proposed representation is very useful for many tasks within the field of tracking and mapping using depth data. In particular, it can complement deep learning approaches for object-wise shape completion [7], or the ScanComplete system [8]: while the network predicts meaningful geometry and semantics, our formulation can constrain camera tracking and denoise the final geometry, taking into account the given semantics.

Acknowledgements

This work was supported by the ERC Consolidator Grant "3D Reloaded".

References

[1] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2008. 2, 4

[2] E. Bylow, J. Sturm, C. Kerl, F. Kahl, and D. Cremers. Real-time camera tracking and 3D reconstruction using signed distance functions. In *Robotics: Science and Systems (RSS)*, 2013. 2, 5, 6, 7

[3] S. Choi, Q. Zhou, and V. Koltun. Robust reconstruction of indoor scenes. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 7

[4] D. Cremers. Dynamical statistical shape priors for level set based tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 28(8):1262–1273, 2006. 3

[5] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *ACM SIGGRAPH, Conference on Computer Graphics and Interactive Techniques*, 1996. 2

[6] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt. BundleFusion: Real-time globally consistent 3D reconstruction using on-the-fly surface re-integration. *ACM Transactions on Graphics (TOG)*, 36(3), 2017. 7

[7] A. Dai, C. R. Qi, and M. Nießner. Shape completion using 3D-encoder-predictor CNNs and shape synthesis. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 8

[8] A. Dai, D. Ritchie, M. Bokeloh, S. Reed, J. Sturm, and M. Nießner. ScanComplete: Large-scale scene completion and semantic segmentation for 3D scans. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 8

[9] M. Dou, L. Guan, J.-M. Frahm, and H. Fuchs. Exploring high-level primitives for indoor 3D reconstruction with a hand-held RGB-D camera. In *Asian Conference on Computer Vision (ACCV)*, 2012. 2

[10] M. Dzitsiuk, J. Sturm, R. Maier, L. Ma, and D. Cremers. Denoising, stabilizing and completing 3D reconstructions on-the-go using plane priors. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2017. 2, 7

[11] F. Engelmann, J. Stückler, and B. Leibe. Joint object pose estimation and shape reconstruction in urban street scenes using 3D shape priors. In *German Conference on Pattern Recognition (GCPR)*, 2016. 3

[12] C. Engels, H. Stewénius, and D. Nistér. Bundle adjustment rules. *Photogrammetric Computer Vision*, 2(32), 2006. 5

[13] N. Fioraio and L. D. Stefano. Joint detection, tracking and mapping by semantic bundle adjustment. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 2

[14] A. Handa, T. Whelan, J. McDonald, and A. J. Davison. A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2014. 1, 7, 8

[15] C. Hertzberg, R. Wagner, U. Frese, and L. Schröder. Integrating generic sensor fusion algorithms with sound state representations through encapsulation of manifolds. *Information Fusion*, 14(1):57–77, 2013. 2, 3

[16] J. Huang, A. Dai, L. Guibas, and M. Nießner. 3DLite: Towards commodity 3D scanning for content creation. *ACM Transactions on Graphics (TOG)*, 36(6), 2017. 2

[17] D. A. Klain and G.-C. Rota. *Introduction to Geometric Probability*. Cambridge University Press, 1997. 3

[18] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. g2o: A general framework for graph optimization. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2011. 2, 5

[19] M. E. Leventon, W. E. L. Grimson, and O. Faugeras. Statistical shape influence in geodesic active contours. In *IEEE EMBS International Summer School on Biomedical Imaging*, 2002. 3

[20] C. Loop. Smooth subdivision surfaces based on triangles. Master's thesis, Department of Mathematics, The University of Utah, 1987. 6

[21] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *ACM SIGGRAPH, Conference on Computer Graphics and Interactive Techniques*, 21(4):163–169, 1987. 6

[22] L. Ma, C. Kerl, J. Stückler, and D. Cremers. CPA-SLAM: Consistent plane-model alignment for direct RGB-D SLAM. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2016. 2

[23] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2011. 2

[24] C. V. Nguyen, S. Izadi, and D. Lovell. Modeling Kinect sensor noise for improved 3D reconstruction and tracking. In *International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission (3DIMPVT)*, 2012. 4, 7

[25] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger. Real-time 3D reconstruction at scale using voxel hashing. *ACM Transactions on Graphics (TOG)*, 32(169), 2013. 2, 8

[26] S. Osher and R. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*. Springer New York, 2003. 2

[27] G. Rudolph and M. Schmidt. *Differential Geometry and Mathematical Physics*. Springer Netherlands, 2013. 2

[28] S. Rusinkiewicz, O. Hall-Holt, and M. Levoy. Real-time 3D model acquisition. *ACM Transactions on Graphics (TOG)*, 21(3):438–446, 2002. 2

[29] R. B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (FPFH) for 3D registration. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2009. 6

[30] R. B. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*, 2011. 6

[31] R. F. Salas-Moreno, B. Glocken, P. H. Kelly, and A. J. Davison. Dense planar SLAM. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2014. 2, 7, 8

[32] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison. SLAM++: Simultaneous localisation and mapping at the level of objects. In *IEEE Interna-*

tional Conference on Computer Vision and Pattern Recognition (CVPR), 2013. 2, 8

[33] R. Schnabel, R. Wahl, and R. Klein. Efficient RANSAC for point-cloud shape detection. *Computer Graphics Forum (CGF)*, 26(2):214–226, 2007. 6

[34] M. Slavcheva, W. Kehl, N. Navab, and S. Ilic. Sdf-2-sdf: Highly accurate 3D object reconstruction. In *European Conference on Computer Vision (ECCV)*, 2016. 7, 8

[35] F. Steinbrücker, J. Sturm, and D. Cremers. Volumetric 3D mapping in real-time on a CPU. In *IEEE International Conference on Robotics and Automation ICRA*, 2014. 2

[36] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of RGB-D SLAM systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012. 7

[37] Y. Taguchi, Y.-D. Jian, S. Ramalingam, and C. Feng. SLAM using both points and planes for hand-held 3D sensors. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2012. 2

[38] A. J. B. Trevor, J. G. Rogers, and H. I. Christensen. Planar surface SLAM with 3D and 2D sensors. In *IEEE International Conference on Robotics and Automation ICRA*, 2012. 2

[39] Y. Zhang, W. Xu, Y. Tong, and K. Zhou. Online structure analysis for real-time indoor scene reconstruction. *ACM Transactions on Graphics (TOG)*, 34(5), 2015. 2