

# SDF-2-SDF: Highly Accurate 3D Object Reconstruction

Miroslava Slavcheva<sup>1,2</sup>(✉), Wadim Kehl<sup>1</sup>, Nassir Navab<sup>1</sup>, and Slobodan Ilic<sup>1,2</sup>

<sup>1</sup> Technische Universität München, Munich, Germany  
{mira.slavcheva,nassir.navab}@tum.de, {kehl,slobodan.ilic}@in.tum.de  
<sup>2</sup> Siemens AG, Munich, Germany

**Abstract.** This paper addresses the problem of 3D object reconstruction using RGB-D sensors. Our main contribution is a novel implicit-to-implicit surface registration scheme between signed distance fields (SDFs), utilized both for the real-time frame-to-frame camera tracking and for the subsequent global optimization. SDF-2-SDF registration circumvents expensive correspondence search and allows for incorporation of multiple geometric constraints without any dependence on texture, yielding highly accurate 3D models. An extensive quantitative evaluation on real and synthetic data demonstrates improved tracking and higher fidelity reconstructions than a variety of state-of-the-art methods. We make our data publicly available, creating the first object reconstruction dataset to include ground-truth CAD models and RGB-D sequences from sensors of various quality.

**Keywords:** Object reconstruction · Signed distance field · RGB-D sensors

## 1 Introduction

The persistent progress in RGB-D sensor technology has prompted exceptional focus on 3D object reconstruction. A key research goal is recovering the geometry of a static target from a moving depth camera. This entails estimating the device motion and fusing the acquired range images into consistent 3D models. Depending on the particular task, methods differ in their speed, accuracy and generality. Most existing solutions are SLAM-like, thus their applications lie in the field of robotic navigation where precise reconstructions are of secondary importance. In contrast, the growing markets of 3D printing, reverse engineering, industrial design, and object inspection require rapid prototyping of high quality models, which is the aim of our system.

One of the most influential works capable of real-time reconstruction is KinectFusion [12, 24]. It conveniently stores the recovered geometry in an incrementally built signed distance field (SDF). However, its frame-to-model camera tracking via iterative closest points (ICP [1, 6]) limits it to objects with distinct

---

**Electronic supplementary material** The online version of this chapter (doi:[10.1007/978-3-319-46448-0\\_41](https://doi.org/10.1007/978-3-319-46448-0_41)) contains supplementary material, which is available to authorized users.



**Fig. 1.** SDF-2-SDF reconstructions of the proposed dataset objects. Colors vary due to difference between synthetic rendering and 3D-printed models, and camera radiometrics (Color figure online)

appearance and to uniform scanning trajectories. Other techniques use a point-to-implicit scheme [3, 5] that avoids explicit correspondence search by directly aligning the point clouds of incoming depth frames with the growing SDF. Such registration has proven to be more robust than ICP, but becomes unreliable when range data is sparse or once the global model starts accumulating errors. Dense visual odometry (DVO) [15] is a SLAM approach that combines image intensities with depth information for warping between RGB-D frames in a Lucas-Kanade-like fashion [20]. Although it is susceptible to drift on poorly textured scenes, DVO achieves impressive accuracy in real time and has been incorporated as the tracking component of the object reconstruction pipeline of Kehl *et al.* [13]. The final step of the latter is a  $g^2o$  pose graph optimization [17] to ensure optimal alignment between all views. While it leads to improved model geometry, it might become prohibitively expensive for a larger amount of keyframes.

Addressing these limitations, we present SDF-2-SDF, a highly accurate 3D object reconstruction system. It comprises online frame-to-frame camera tracking, followed by swift multi-view pose optimization during the generation of the output reconstruction. These two stages can be used as completely stand-alone tools. Both of them employ the SDF-2-SDF registration method, which directly minimizes the difference between pairs of SDFs. Moreover, its formulation allows for integration of surface normal information for better alignment. In addition to handling larger motion and no dependence on texture, our frame-to-frame tracking strategy avoids drift caused by errors in the global model. Finally, our global refinement is faster than the pose graph optimization used in other pipelines [10, 13, 17]. Tackling the lack of a dataset combining ground-truth CAD models and RGB-D sequences with known camera trajectories, we have created a 3D-printed object dataset, which we make publicly available (*cf.* Sect. 5.1 for details). We highlight our contributions below:

- precise implicit-to-implicit registration between SDFs for online frame-to-frame camera tracking,
- introduction of a global pose optimization step, which is elegantly interleaved with the model reconstruction,
- improved convergence via incorporation of surface orientation constraints,
- the first object reconstruction dataset including both ground-truth 3D models and RGB-D data from sensors of varying quality.

Our parallel tracking implementation runs in real-time on the CPU. While pose refinement is only essential for low-quality depth input, it is interleaved with the final model computation, adding just a few seconds of processing. Sample outputs of our pipeline are displayed in Fig. 1.

We performed exhaustive evaluation on synthetic and real input. Moreover, the tracking precision of SDF-2-SDF without refinement was compared with Generalized-ICP [33], KinectFusion [24], point-to-implicit methods [3, 5] and DVO [15]. The fidelity of the non-optimized model was assessed against KinectFusion, while the refined reconstruction was compared to that of Kehl *et al.*'s pipeline [13] that includes posterior optimization.

## 2 Related Work

Fully automatic object reconstruction requires knowing the precise 6 degrees-of-freedom camera poses from which the RGB-D views were obtained. Arguably, the most widespread strategy for aligning depth data is ICP [1, 6, 31]. Although it is simple and generic, the method performs poorly in the presence of gross statistical outliers and large motion. Moreover, it is rather costly because it involves recomputing point correspondences in every iteration.

KinectFusion [12, 24] employs an implicit surface representation (an SDF) for the continuously incremented reconstruction, but ray traces it into a point cloud on which a multi-scale point-to-plane ICP is used for frame-to-model registration. Thus, it suffers from the related drawbacks. Through a comparison to PCL's implementations of GICP [33] and KinFu [27], we show that SDF-2-SDF can handle cases where ICP fails.

Several authors [3, 5, 7, 16, 22, 26, 30, 40] report superior registration using implicit surface representations. Notably, Bylow *et al.* [3] and Canelhas *et al.* [5] directly project the points of a tracked frame onto a global cumulative SDF and reduce the registration problem to solving an inexpensive  $6 \times 6$  equation system in every iteration. Similar to [37] who leverage point-to-NDT (normal distribution transform) to NDT-to-NDT, we extend the point-to-implicit strategy to aligning pairs of SDFs. Comparisons indicate higher precision of SDF-2-SDF thanks to reduced effect of the noise inherent to explicit point coordinates.

A fast and powerful tracking system that works well on textured scenes is DVO [15, 36]. It employs a photo-consistency constraint to find the best alignment between two RGB-D frames. Despite requiring a polychromatic support for the object of interest, visual odometry is used in the reconstruction pipelines of Dimashova *et al.* [10] and Kehl *et al.* [13]. These two works then execute a  $g^2o$  pose graph optimization [17], which undoubtedly yields higher quality meshes,

but is rather costly. When used in dense scene reconstruction applications, graph-based optimization may last hours to days [44]. We propose improving the estimated trajectory via global implicit-to-implicit optimization. Selected keyframes are re-registered to a global SDF weighted average, which can be readily used as output reconstruction, making our refinement significantly faster.

Industrial scanning scenarios often do not permit augmenting the scene with textured components that aid tracking. Therefore we have designed an entirely photometry-independent system. Nevertheless, our energy formulation allows for combining a multitude of geometric constraints. Masuda [22] uses the difference between normal vectors to robustify SDF registration. Instead, we take the dot product as a more accurate measure of surface orientation similarity. While our approach works well without these additional constraints, they are straightforward to integrate into the SDF-2-SDF framework and lead to faster convergence.

A thorough evaluation of our system requires both ground-truth trajectories and object models. The TUM RGB-D benchmark [38] includes an ample set of sequences with associated poses, while the ICL-NUIM dataset [11] provides the synthetic model of one scene. However, both are intended for SLAM applications and feature large spaces rather than smaller-scale objects. Similarly, point cloud benchmarks [28, 29] cannot be used directly by methods designed for range image registration. Existing RGB-D collections of household items, such as that of Washington University [18] and Berkeley’s BigBIRD [34], either lack noiseless meshes or complete 6 DoF poses [23]. Therefore we 3D-printed a selection of objects with different geometry, size and colors, and contribute the first, to the best of our knowledge, object dataset with original CAD models and RGB-D data from various quality sensors, acquired from externally measured trajectories.

### 3 Geometric Preliminaries

This section describes the specifics of the SDF generation approach we used.

#### 3.1 Mathematical Notation

An RGB-D sensor delivers a pair consisting of a depth map  $D : \mathbb{N}^2 \rightarrow \mathbb{R}$  and a corresponding color image  $I : \mathbb{N}^2 \rightarrow \mathbb{R}^3$ . Given a calibrated device, the projection  $\pi : \mathbf{x} = \pi(\mathbf{X})$  maps a 3D point  $\mathbf{X} = (X, Y, Z)^\top \in \mathbb{R}^3$  onto the image location  $\mathbf{x} = (x, y)^\top \in \mathbb{N}^2$ . The inverse relation  $\pi^{-1}$  determines the 3D coordinates  $\mathbf{X}$  from a pixel  $\mathbf{x}$  with depth value  $D(\mathbf{x})$  in a range image:  $\mathbf{X} = \pi^{-1}(\mathbf{x}, D(\mathbf{x}))$ .

The registration problem requires determining the rigid body transformation between the camera poses from which a pair of images were acquired. This 6 degree-of-freedom motion consists of a rotation  $\mathbf{R} \in SO(3)$  and a translation  $\mathbf{t} \in \mathbb{R}^3$ . We use twist coordinates from the Lie algebra  $se(3)$  of the special Euclidean group  $SE(3)$ , as they provide a minimal representation of the motion [21]:

$$\xi = (\mathbf{u} \ \boldsymbol{\omega})^\top = (u_1, u_2, u_3, \omega_1, \omega_2, \omega_3)^\top, \quad (1)$$

where  $\boldsymbol{\omega} \in \mathbb{R}^3$  corresponds to the rotational component and  $\mathbf{u} \in \mathbb{R}^3$  stands for the translation. We denote the motion of any 3D point  $\mathbf{X}$  in terms of  $\xi$  as  $\mathbf{X}(\xi)$ .

### 3.2 Signed Distance Fields

An SDF in 3D space is an implicit function  $\phi: \Omega \subseteq \mathbb{R}^3 \rightarrow \mathbb{R}$  that associates each point  $\mathbf{X} \in \mathbb{R}^3$  with the signed distance to its closest surface location [25]. Points within the object bounds have negative signed distance values, while points outside have positive values. Their interface is the object surface, which can be extracted as the zeroth level-set crossing via marching cubes or ray tracing. As implicit functions, SDFs have smoothing properties that make them superior to explicit 3D coordinates in many applications, as we will see in the comparison between point-to-implicit and implicit-to-implicit registration.

Registration is done by aligning projective truncated SDFs. First, the bounding volume is discretized into cubic voxels of predefined side length  $l$ .

A point  $\mathbf{X}$  belongs to the voxel with index  $vox: \mathbb{R}^3 \rightarrow \mathbb{N}^3$ :

$$vox(\mathbf{X}) = \text{int} \left( 1/l(\mathbf{X} - \mathbf{C}) - (1/2, 1/2, 1/2)^\top \right), \quad (2)$$

where  $\text{int}$  rounds to integers, and  $\mathbf{C}$  is the lower-left corner of the volume. All points within the same voxel are assigned the properties of its center

$$\mathbf{V}(\mathbf{X}) = l(vox(\mathbf{X}) + (1/2, 1/2, 1/2)^\top) + \mathbf{C}, \quad (3)$$

so we use  $\mathbf{V}$  to denote the entire voxel. As a range image only contains measurements of surface points, the signed distance is the difference of sensor reading for the voxel center projection  $\pi(\mathbf{V})$  and its depth  $\mathbf{V}_Z$ :

$$\phi_{true}(\mathbf{V}) = D(\pi(\mathbf{V})) - \mathbf{V}_Z \quad (4)$$

$$\phi(\mathbf{V}) = \begin{cases} \text{sgn}(\phi_{true}(\mathbf{V})) & , \text{ if } |\phi_{true}(\mathbf{V})| \geq \delta \\ \phi_{true}(\mathbf{V})/\delta & , \text{ otherwise} \end{cases} \quad (5)$$

$$\omega(\mathbf{V}) = \begin{cases} 1 & , \text{ if } \phi_{true}(\mathbf{V}) > -\eta \\ 0 & , \text{ otherwise} \end{cases} \quad (6)$$

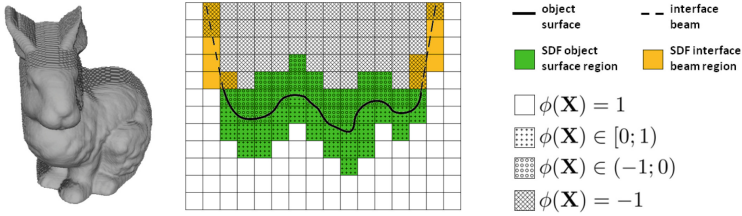
$$\zeta(\mathbf{V}) = I(\pi(\mathbf{V})). \quad (7)$$

The true signed distance value  $\phi_{true}$  is usually scaled by a factor  $\delta$  (related to the sensor error; we used 2 mm) and truncated into the interval  $[-1, 1]$  (Eq. 5).

Binary weights  $\omega$  are associated to the voxels in order to discard unseen areas from computations. All visible locations and a region of size  $\eta$  behind the surface (reflecting the expected object thickness) are assigned weight one (Eq. 6).

Finally, in order to be able to assign colors to the output mesh, we store a corresponding RGB triple for every voxel in an additional grid  $\zeta$  of the same dimensions as  $\phi$  (Eq. 7).

This approach of SDF generation from a single frame creates *beams* on the *interface* formed by the camera ray through the surface silhouette: voxels outside the object have signed distance 1 and neighbour with voxels behind the surface with value  $-1$  (cf. Fig. 2). Beams cancel out when SDFs from multiple viewpoints are fused, and are easily omitted from calculations on projective SDFs, since



**Fig. 2.** Single-frame projective truncated SDF: marching cubes rendering, exhibiting viewpoint-dependent interface beams (left); cross-section along the x-y plane of the TSDF volume, identifying specialized regions (middle, right)

all voxels behind the surface have not been observed and have weight zero. However, beam voxels have faulty gradient values, but are easily excluded as well: a gradient computed via central differences will have at least one component with absolute value 1. We will often use SDF gradients, since the normalized 3D spatial gradient  $\nabla_{\mathbf{X}}\phi$  equals the normals  $\bar{\mathbf{n}}$  at surface locations [25].

An important Jacobian that will be needed in the numeric schemes is obtained when deriving point coordinates  $\mathbf{X} \in \mathbb{R}^3$  with respect to a pose  $\xi$ . After applying the chain rule  $\nabla_{\xi}\phi(\mathbf{X}(\xi)) = \nabla_{\mathbf{X}}\phi(\mathbf{X})\frac{\partial\mathbf{X}}{\partial\xi}$ , the following holds:

$$\nabla_{\xi}\phi(\mathbf{X}) = \nabla_{\mathbf{X}}\phi(\mathbf{X}) (\mathbf{I}_{3\times 3} \mid -(\mathbf{X}(\xi^{-1}))_{\times}) \in \mathbb{R}^{1\times 6}. \tag{8}$$

Once the camera poses have been determined, their SDFs can be fused into a common model  $\Phi$  via the weighted averaging scheme of Curless and Levoy [9]:

$$\begin{aligned} \Phi_{t+1}(\mathbf{V}) &= \frac{W_t(\mathbf{V})\Phi_t(\mathbf{V}) + \omega_{t+1}(\mathbf{V})\phi_{t+1}(\mathbf{V})}{W_t(\mathbf{V}) + \omega_{t+1}(\mathbf{V})}, \\ W_{t+1}(\mathbf{V}) &= W_t(\mathbf{V}) + \omega_{t+1}(\mathbf{V}). \end{aligned} \tag{9}$$

Each color grid channel is averaged similarly. The color weights at every voxel equal the product of  $\omega$  and the cosine of the viewing ray angle, so that points whose normal is oriented towards the camera have stronger influence [3].

Note that color and normals are only valid at surface locations, defined as the ca. 1–5 voxel-wide narrow band of non-truncated voxels (*cf.* Fig. 2). This is an apt surface approximation permitting fast binary checks:  $|\phi(\mathbf{V})| < 1$ .

## 4 SDF-2-SDF Registration

Our system takes a stream of RGB-D data and pre-processes it by masking the object of interest as done in [13, 32] and optionally de-noising the depth images via anisotropic diffusion [41] or bilateral filtering [39]. The frame bounding volume is automatically estimated by back-projection of all depth map pixels, as opposed to model-based methods that require manual volume selection. The volume is then slightly padded and used for the generation of both SDFs that are currently being aligned.

These steps are applied to each input image fed to our tracking method which performs frame-to-frame SDF-2-SDF registration, thus avoiding error accumulation and allowing for a moving volume of interest. Once tracking is complete, a predefined number of keyframes are globally SDF-2-SDF-registered to their weighted average. This refinement follows a coarse-to-fine scheme over voxel size. Finally, a colored surface mesh is obtained via the marching cubes algorithm [19].

#### 4.1 Objective Function

We propose a simple scheme for implicit-to-implicit registration, whereby the direct difference of two SDFs is iteratively minimized. To achieve best alignment between frames, their per-voxel difference has to be minimal: truncated voxels have the same values, while the near-surface non-truncated voxels from both grids steer convergence towards surface overlap. Registration is facilitated by the fact that both SDFs encode the distance to the common surface. In the experimental evaluation we will demonstrate that this leads to more accurate results than when one frame is explicitly represented by its 3D cloud.

In the following  $\phi_{ref}$  is the reference, while  $\phi_{cur}(\xi)$  is the SDF of the current frame, whose pose  $\xi^*$  we are seeking. Since all voxels  $\mathbf{V}$  are affected by the same transformation, their contributions can be straightforwardly added up. To ease notation, when summing over all voxels, we will omit the coordinates, e.g. we will write  $\phi_{ref}$  instead of  $\phi_{ref}(\mathbf{V})$  in sums. The main signed distance energy component is  $E_{geom}$ , to which surface normal constraints  $E_{norm}$  can be added, yielding the SDF-2-SDF objective function  $E_{SDF}$ :

$$E_{SDF}(\xi) = E_{geom}(\xi) + \alpha_{norm} E_{norm}(\xi) , \quad (10)$$

$$E_{geom}(\xi) = \frac{1}{2} \sum_{voxels} \left( \phi_{ref} \omega_{ref} - \phi_{cur}(\xi) \omega_{cur}(\xi) \right)^2 , \quad (11)$$

$$E_{norm}(\xi) = \sum_{surface\ voxels} \left( 1 - \bar{\mathbf{n}}_{ref} \cdot (\bar{\mathbf{n}}_{cur}(\xi)) \right) . \quad (12)$$

The influence of  $E_{norm}$  is adjusted through its weight. By default it is not used in order to ensure optimal speed. For the tests with curvature constraints, we empirically found values in the range  $(0, 1]$  to be reliable for  $\alpha_{norm}$ .

#### 4.2 Camera Tracking

Frame-to-model tracking can be detrimental in object reconstruction, since errors in pose estimation can introduce incorrect geometry when fused into the global model and adversely affect the subsequent tracking. Therefore we favor frame-to-frame camera tracking on single-frame SDFs.

We determine the relative transformation between two RGB-D frames by setting the pose of the first one to identity and incrementally updating the other one. The tracking minimization scheme for the geometry term is based on a first-order Taylor approximation around the current pose estimate  $\xi^k$  (Eqs. 13–15).

Like several related approaches, it leads to an inexpensive  $6 \times 6$  linear system (Eq. 16). Weighting terms have been omitted from formulas for clarity. In order to avoid numerical instability, we take a step of size  $\beta$  towards the optimal solution (Eq. 17). In each iteration  $\phi_{cur}$  is generated from the current pose estimate. We terminate when the translational update falls below a threshold [38].

$$\mathbf{A} = \sum_{\text{voxels}} \nabla_{\xi}^{\top} \phi_{cur}(\xi^k) \nabla_{\xi} \phi_{cur}(\xi^k) \quad (13)$$

$$\mathbf{b} = \sum_{\text{voxels}} \left( \phi_{ref} - \phi_{cur}(\xi^k) + \nabla_{\xi} \phi_{cur}(\xi^k) \xi^k \right) \nabla_{\xi}^{\top} \phi_{cur}(\xi^k) \quad (14)$$

$$\frac{dE_{geom}}{d\xi} = \mathbf{A}\xi - \mathbf{b} \quad (15)$$

$$\xi^* = \mathbf{A}^{-1}\mathbf{b} \quad (16)$$

$$\xi^{k+1} = \xi^k + \beta(\xi^* - \xi^k) \quad (17)$$

As stated in the preliminary section, the normals of the SDF equal its spatial gradient. Therefore, the surface orientation term imposes curvature constraints, whose derivation is mathematically equivalent to a second-order Taylor approximation of  $E_{geom}$ . Thus the objective remains the same, but convergence is speeded up. We obtain the following formula for the derivative of  $E_{norm}$  with respect to each component  $j$  of the twist coordinates:

$$\frac{dE_{norm}}{d\xi_j} = \sum_{\text{surface voxels}} -\bar{\mathbf{n}}_{ref} \cdot \left( (\nabla_{\mathbf{x}} \bar{\mathbf{n}}_{cur}(\xi)) (\mathbf{I}_{3 \times 3} \mid -(\mathbf{V}(\xi^{-1}))_{\times}) \delta_j \right), \quad (18)$$

where  $\delta_j$  is a 6-element vector of zeros with  $j$ -th component 1.

### 4.3 Global Pose Optimization

After tracking, a predefined number of regularly spaced keyframes are taken for generation of the final reconstruction. The weighted averaging provides a convenient way to incorporate the information from all of their viewpoints into a global model. However, when using noisy data the estimated trajectory might have accumulated drift, so the keyframes' poses need to be refined to ensure optimal geometry. For this task we propose a frame-to-model scheme based on the SDF-2-SDF registration energy, in which each pose  $\xi_p$  is better aligned with the global weighted average model. In effect, the optimization is interleaved with the computation of the final reconstruction, and only takes less than half a minute. The linearization of the energy follows a gradient descent minimization:

$$\frac{dE_{geom}}{d\xi} = \sum_{\text{voxels}} (\phi_{cur}(\xi) - \phi_{avg}) \nabla_{\xi} \phi_{cur}(\xi), \quad (19)$$

$$\xi_p^{k+1} = \xi_p^k - \alpha \frac{dE_{geom}(\xi_p^k)}{d\xi}. \quad (20)$$



The pose of the first camera is used as a reference and is fixed to identity throughout the whole optimization. In each iteration, the pose updates of all others are determined relative to the global model, after which they are simultaneously applied. The weighted average is recomputed every couple of iterations (10 in our case), so that the objective does not change in the meantime. Furthermore, this is done in a coarse-to-fine scheme over the voxel size to ensure that larger pose deviations can also be recovered.

## 5 Experimental Evaluation

In the following we compare our method to state-of-the-art approaches on data acquired with different RGB-D sensors. In all scenarios we assume a single rigid object of interest. Since the two stages of our pipeline can be used stand-alone, we assess tracking and reconstruction separately. The presented results employ solely the geometric component of the objective function, unless stated otherwise.

### 5.1 Test Set-up and Dataset

In order to thoroughly assess our system, we selected a set of **5 CAD models** of objects exhibiting various richness of geometry and texture (shown in Fig. 1): uni-colored (*bunny*), colored in patches (*teddy*, *Kenny*), densely colored (*leopard*, *tank*); thin (*leopard*), very small (*Kenny*), very large (*teddy*), with spherical components (*teddy*, *Kenny*).<sup>1</sup> They were 3D printed in color with a *3D Systems ZPrinter 650*, which reproduces details of resolution 0.1 mm [43]. In this way we ensure that the textured *groundtruth models* are at our disposal for evaluation, and eliminate dependence on the precision of a stitching method or system calibration that existing datasets entail.

We acquired **3 levels of RGB-D data accuracy** for each of these models: noise-free synthetic rendering in *Blender* [2], industrial phase shift sensor of resolution 0.13 mm, and a Kinect v1. We followed **2 scanning modes: turntable and handheld** for synthetic and Kinect data. Each of the synthetically generated trajectories has radius 50 cm and includes 120 poses, the handheld one is a sine wave with frequency 5 and amplitude 15 cm. The *camera poses* for the Kinect are known through a markerboard placed below the object of interest. The industrial sensor takes 4 s to acquire an RGB-D pair, permitting us to only record turntable sequences. Due to its limited field of view, we could not place a sufficiently large markerboard, so we will only use it for evaluation of model accuracy. In all cases the object of interest is put on a textured support that ensures optimal conditions for visual odometry.

Summing up, we aimed to cover a wide range of object scanning scenarios with this evaluation setup. The used RGB-D sequences for every object, sensor and mode, together with the respective ground-truth CAD models and camera trajectories are available at <http://campar.in.tum.de/personal/slavcheva/3d-printed-dataset/index.html>.

<sup>1</sup> The *bunny* is from The Stanford Repository [35], all other models are freely available at <http://archive3d.net/>. Object sizes are listed in the supplementary material.

## 5.2 Trajectory Accuracy

*SDF-2-SDF* tracking was compared with the following:

- **ICP-based approaches:** PCL’s [27] frame-to-frame generalized ICP [33] (*GICP*) and frame-to-model KinectFusion [24] (*KinFu*);
- **point-to-implicit methods:** frame-to-model [5] (*FM-pt-SDF*, available as a ROS package [4]) and our frame-to-frame modification (*FF-pt-SDF*);
- **visual odometry:** DVO [15] (available online [14]) only on the object (*DVO-object*) and on the object and its richly textured background (*DVO-full*).

We evaluate the **relative** pose error (RPE) [38] per frame transformation and take the root-mean-squared, average, minimum and maximum:

$$RPE_{i \rightarrow i+1} = (\mathbf{P}_i^{-1} \mathbf{P}_{i+1})^{-1} (\mathbf{Q}_i^{-1} \mathbf{Q}_{i+1}), \quad (21)$$

where  $i$  is the frame number,  $\{\mathbf{Q}_{1..n}\}$  is the groundtruth trajectory and  $\{\mathbf{P}_{1..n}\}$  is the estimated one. It evaluates the difference between the groundtruth and estimated *transformations*, and equals identity when they are perfectly aligned.

In addition, we report the average, minimum and maximum angular error per transformation. All errors are also evaluated for the **absolute** poses. Note that while our relative metric is equal to the RGB-D benchmark RPE per frame, our absolute metric is, in general, more severe than its absolute trajectory error [38]. This is because the ATE targets SLAM and first finds the best alignment between trajectories, while we use the same initial reference pose for both trajectories, since this directly influences the way frames are fused into meshes. To economize on space, we plot only the average errors, which we found to be most conclusive, and refer the reader to the supplementary material for a full tabular overview.

*Synthetic data.* As a proof of concept, we run tests on synthetic data, the details of which can be found in the supplementary material. SDF-2-SDF clearly outperforms the other methods with an average relative drift below 0.4 mm and angular error below  $0.06^\circ$ . The average absolute pose error of 2 mm corresponds to the used voxel size and suggests that given good data, only the grid resolution limits our tracking accuracy. Notably, it performs equally well regardless of object geometry and yields a negligible error with respect to the trajectory size.

*Kinect sequences.* Figure 3 indicates that GICP and DVO using only the object perform worst, while DVO using the object and its background performs best. SDF-2-SDF outperforms the remaining methods and is even more precise than DVO-full on turntable *bunny*, *teddy*, *tank* and handheld *teddy*, despite using only geometric constraints on the object of interest. KinFu and the two pt-SDF strategies perform similar to each other. In most cases KinFu is more accurate than pt-SDF, while frame-to-frame is slightly better than the frame-to-model pt-SDF variant. A notable failure case for FM-pt-SDF was the turntable *teddy*, where symmetry on the back caused drift from the middle of the sequence onwards, which lead to unrepairable errors in the global model and consequently flawed tracking. Similarly, FM-pt-SDF performed poorly on the turntable *Kenny* due

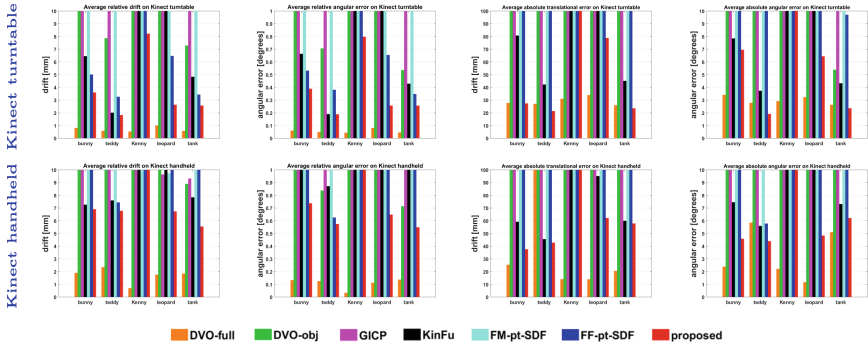


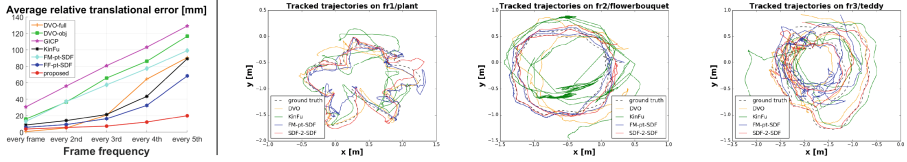
Fig. 3. Comparison of tracking errors on Kinect data

to its fine structures, while FF-pt-SDF did not suffer from error build-up and was most accurate. These observations lead to the conclusion that a frame-to-frame strategy is better for object reconstruction scenarios, where, as opposed to SLAM, there is no repeated scanning of large areas that could aid tracking. Thus, we also chose a frame-to-frame tracking mode for SDF-2-SDF, and achieve higher precision than pt-SDF due to the denser formulation of our error function.

In the presence of severe Kinect-like noise, the sparse point clouds of objects in the scene tend to become too corrupted and degrade registration accuracy. On the contrary, SDF-2-SDF is more precise than cloud-based KinFu and pt-SDF because the inherent smoothing properties of volumetric representations handle noise better. Moreover, SDF-2-SDF relies on a denser set of correspondences: on average, the used clouds consist of  $8 \cdot 10^3$  data points, while the SDFs have  $386 \cdot 10^3$  voxels. Thus the problem is constrained more strongly, making our proposed registration strategy more suitable for object reconstruction.

*Contribution of surface orientation constraints.* As  $E_{norm}$  is a second-order energy term, it does not significantly influence accuracy, but rather convergence speed. With  $\alpha_{norm} = 0.1$  registration is 3–6% more precise and takes 1.5–2.2 times less iterations. However, processing normals increases the time per iteration by 30–45%. Thus  $E_{norm}$  is beneficial for objects of distinct geometry, where normals can be reliably estimated, or when a low-noise sensor is used. For consistency and real-time speed, elsewhere in the paper we evaluate only on  $E_{geom}$ .

*Convergence Analysis.* We empirically analyzed the convergence basin of the available registration methods. To mimic initial conditions where the global minimum is further away, we tested by skipping frames from the Kinect turntable sequences. Figure 4(left) contains the averaged results. The error of SDF-2-SDF grows at the slowest rate, indicating that thanks to its denser set of correspondences, it can determine an accurate pose from a much larger initial deviation (up to ca.  $15^\circ$ ), i.e. it can cope with smaller overlap. Notably, when taking every third frame or fewer, SDF-2-SDF is considerably more precise than DVO-full.



**Fig. 4.** Convergence analysis of registration methods with respect to frame distance (left). Estimated trajectories on the TUM RGB-D benchmark [38] (right)

*Other Public Data.* While our goal is object reconstruction rather than SLAM, we also evaluated the tracking component of our system on several sequences of the *3D Object Reconstruction* category of the TUM RGB-D benchmark [38]. They contain moderately cluttered scenes, unconstrained camera motion and occasionally missing depth data due to close proximity to the sensor. The whole images were used for DVO and GICP, while all other methods tracked solely using the bounding volume of the object of interest. Nevertheless, our SDF-2-SDF was the most precise method on *fr1/plant* and *fr3/teddy*, and was only slightly less accurate than FM-pt-SDF on *fr2/flowerbouquet*. The reason is that its leaves have no effective thickness, therefore the SDFs lose their power in discerning inside from outside and, depending on parameters, might oversmooth and become inferior to point cloud registration. This effect can be mitigated by a finer voxel size, at the cost of slower processing. Figure 4(right) summarizes the results, while numerical details can be found in the supplementary material.

### 5.3 3D Model Accuracy

As our ultimate goal is highly accurate 3D reconstruction, we assess the fidelity of output meshes against their CAD models. We report the cloud-to-model absolute distance mean and standard deviation, measured in *CloudCompare* [8].

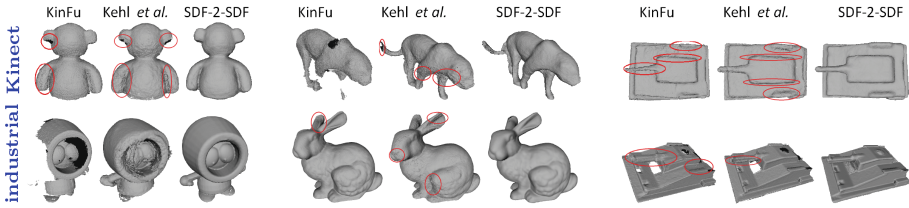
First, the mesh obtained at the end of tracking is compared with that of *KinFu*, juxtaposing two real-time methods that do not employ posterior pose optimization (blue rows in Table 1).

Next, the mesh yielded after global refinement is compared to the method of Kehl *et al.* [13], which tracks by DVO [15], detects loop closure, optimizes the poses of 30 keyframes via  $g^2o$  [17], and integrates them via Zach *et al.* [42]’s TV- $L^1$  scheme. The code was kindly provided to us by the authors. To highlight the dependence of odometry on texture, we evaluated both on the object with its textured support (Kehl *et al.* full) and on the object only (Kehl *et al.* object).

The results in Table 1 indicate that SDF-2-SDF reconstructions are the best for high quality data, regardless whether refinement is carried out. In fact, for most objects the global optimization brings just a small improvement and is therefore not necessary on data with little noise. The mean errors of below 0.3 mm on synthetic and sub-millimeter on industrial are evidence for the high accuracy of SDF-2-SDF meshes. On the contrary, the posterior refinement leads to a considerable (up to 2.5 times) decrease in model error on Kinect data, yielding sub-millimeter

**Table 1.** *CloudCompare* evaluation of the absolute cloud-to-model reconstruction error.

object	method	error [mm]									
		synth. circle		synth. wave		industr. turntab.		Kinect turntable		Kinect handheld	
		mean	std.dev.	mean	std.dev.	mean	std.dev.	mean	std.dev.	mean	std.dev.
bunny	KinFu	0.544	0.677	0.787	0.988	0.664	0.654	3.800	2.840	4.101	3.716
	ours (no refinement)	0.135	0.139	0.133	0.134	0.656	0.438	2.586	1.869	1.770	1.733
	Kehl <i>et al.</i> object	1.459	1.220	4.885	3.732	2.149	2.869	5.156	4.115	8.274	6.013
	Kehl <i>et al.</i> full ours (with refin.)	<b>0.130</b>	<b>0.137</b>	<b>0.131</b>	<b>0.133</b>	<b>0.838</b>	<b>0.860</b>	<b>1.134</b>	<b>1.243</b>	<b>1.124</b>	<b>1.095</b>
teddy	KinFu	0.370	0.275	0.418	0.285	0.998	0.807	1.271	1.045	2.355	1.447
	ours (no refinement)	0.161	0.179	0.146	0.142	0.930	0.588	1.078	0.890	1.589	1.537
	Kehl <i>et al.</i> object	0.358	0.303	0.257	0.193	1.028	0.892	2.306	1.862	2.287	1.826
	Kehl <i>et al.</i> full ours (with refin.)	<b>0.157</b>	<b>0.166</b>	<b>0.146</b>	<b>0.142</b>	<b>4.828</b>	<b>4.215</b>	<b>1.221</b>	<b>0.858</b>	<b>3.066</b>	<b>2.380</b>
Kenny	KinFu	0.418	0.311	0.440	0.359	1.650	1.451	1.511	1.387	2.874	2.727
	ours (no refinement)	0.154	0.151	0.147	0.154	0.363	0.391	1.295	1.311	2.415	2.051
	Kehl <i>et al.</i> object	0.948	0.736	1.931	1.965	1.816	1.710	3.181	3.238	<i>failed</i>	<i>failed</i>
	Kehl <i>et al.</i> full ours (with refin.)	<b>0.152</b>	<b>0.146</b>	<b>0.146</b>	<b>0.150</b>	<b>2.553</b>	<b>2.644</b>	<b>1.263</b>	<b>0.850</b>	<b>2.282</b>	<b>1.381</b>
leopard	KinFu	0.525	0.758	0.540	0.734	1.785	1.299	4.445	2.430	1.886	3.292
	ours (no refinement)	0.226	0.264	0.237	0.268	0.760	0.830	2.692	1.882	1.321	1.220
	Kehl <i>et al.</i> object	0.330	0.324	0.260	0.268	1.018	1.378	5.693	5.050	<i>failed</i>	<i>failed</i>
	Kehl <i>et al.</i> full ours (with refin.)	<b>0.225</b>	<b>0.263</b>	<b>0.233</b>	<b>0.266</b>	<b>3.626</b>	<b>3.705</b>	<b>1.907</b>	<b>1.218</b>	<b>1.281</b>	<b>1.218</b>
tank	KinFu	0.900	0.708	1.274	0.911	1.390	1.315	1.561	1.453	2.579	2.265
	ours (no refinement)	0.270	0.204	0.289	0.263	0.953	0.740	1.336	1.188	2.042	2.404
	Kehl <i>et al.</i> object	0.384	0.506	3.929	3.961	1.573	2.250	1.192	1.009	2.340	2.062
	Kehl <i>et al.</i> full ours (with refin.)	<b>0.267</b>	<b>0.199</b>	<b>0.285</b>	<b>0.263</b>	<b>2.617</b>	<b>2.571</b>	<b>1.064</b>	<b>0.872</b>	<b>0.946</b>	<b>0.806</b>



**Fig. 5.** Qualitative comparisons: Kinect data (top) and industrial sensor (bottom). Symmetry (*teddy*'s and *Kenny*'s backs) and thin structures (*tank*'s gun, *leopard*'s tail and legs) degrade the performance of cloud registration, causing misalignment artifacts (red ellipses), while SDF-2-SDF succeeds (see supplementary for magnified figures) (Color figure online)

precision for many of the objects. Moreover, the error is always below 2 mm, corresponding to the device uncertainty, once again indicating that our method is only limited by the sensor resolution and the chosen voxel size.

Further, the results on Kinect data demonstrate odometry's heavy dependence on texture. Kehl *et al.*'s pipeline even fails for smaller objects such as *leopard* and *Kenny* in the absence of textured surroundings. Not all tests with the industrial sensor followed the same trend, because the provided implementation required resizing the original  $2040 \times 1080$  images to VGA resolution, leading to increased error when processing areas near the image border, where the textured table is located. The results of KinFu and SDF-2-SDF did not change for VGA and the original size, indicating lower sensitivity of volumetric approaches to such issues. Moreover, the speed of SDF-2-SDF remained unaffected, as it only depends on the voxel resolution, and not on image or point cloud size. Thus our system generalizes well not only for various object geometry, but also for any device. Figure 5 shows examples where our denser SDF-2-SDF succeeds, while other methods suffer, both on Kinect and on high-quality data.

**Table 2.** SDF-2-SDF runtime: average/**fastest**/**slowest**

tracking [milliseconds per frame]			refinement [total seconds]		
pre-process	reference SDF generation	minimization iterations	weighted averages	optimizing poses	marching cubes
1.7/ 1.6/ 1.8	2.6/ 1.7/ 3.8	45.3/ 41.4/ 54.9	1.9/ 0.4/ 6.8	6.1/ 0.3/ 20.2	0.6/ 0.2/ 1.3
total 49.6/ 44.7/ 60.5 ms = 20/ 22/ 17 FPS			total 8.6/ 0.9/ 28.3 s		

## 5.4 Implementation Details

We carried out our experiments on an 8-core Intel i7-4900MQ CPU at 2.80 GHz. As tracking at a voxel size finer than the sensor resolution is futile, we used 2 mm for all tests. SSE instructions aid efficient SDF generation, leaving the computation of each voxel’s contribution to the  $6 \times 6$  system as the bottleneck. To speed it up, we do not process voxels which would not have a significant influence, similar to a narrow-band technique: locations with zero weight in either volume and voxels with the same signed distance are disregarded. Thus runtime is not linear in object size, but depends on its geometry. The SDF of an object with bounding cube of side 50 cm requires only 61 MB, so memory is not an issue.

We execute the computations in parallel on the CPU, thereby achieving real-time performance between 17 and 22 FPS. Table 2 lists the time taken for each major step of our pipeline as average over all sequences, as well as the fastest (achieved on *synthetic Kenny*) and slowest (on *Kinect leopard*) runs.

On the other hand, the pose refinement has a simpler mathematical formulation, whereby only a 6-element vector is calculated in every gradient descent step. It requires at most 40 iterations on each voxel resolution level (4 mm, 2 mm, optionally 1 mm), taking up to 30 s to deliver the reconstruction, which is generated via marching cubes from the final field. In comparison, Kehl *et al.*’s pose graph optimization took 196.4 s on average (min 53 s, max 902 s) for the same amount of keyframes. Table 2 shows that our refinement is much faster.

## 6 Conclusions

We have developed a complete pipeline that starts with raw sensor data and delivers a highly precise 3D model without any user interaction. The underlying novel implicit-to-implicit registration method is dense and direct, whereby it avoids explicit correspondence search. The global refinement is an elegant and inexpensive way to jointly optimize the poses of several views and the reconstructed model. Experimental evaluation has shown that our reconstructions are of higher quality than those of related state-of-the-art systems.

**Acknowledgements.** We thank Siemens AG for funding the creation of the 3D printed dataset. We are also grateful to Patrick Wissmann for providing access to the phase shift sensor and to Tolga Birdal for his invaluable help in the image acquisition.

## References

1. Besl, P.J., McKay, N.D.: A method for registration of 3-D shapes. *IEEE Trans. Pattern Anal. Mach. Intell. (PAMI)* **14**(2), 239–256 (1992)
2. Blender Project: Free and open 3D creation software. <https://www.blender.org/>. Accessed 9 Mar 2016
3. Bylow, E., Sturm, J., Kerl, C., Kahl, F., Cremers, D.: Real-time camera tracking and 3D reconstruction using signed distance functions. In: *Robotics: Science and Systems Conference (RSS)* (2013)
4. Canelhas, D.: sdf\_tracker - ROS Wiki. [http://wiki.ros.org/sdf\\_tracker](http://wiki.ros.org/sdf_tracker). Accessed 9 Mar 2016
5. Canelhas, D.R., Stoyanov, T., Lilienthal, A.J.: SDF tracker: a parallel algorithm for on-line pose estimation and scene reconstruction from depth images. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2013)
6. Chen, Y., Medioni, G.: Object modeling by registration of multiple range images. In: *Proceedings of the 1991 IEEE International Conference on Robotics and Automation (ICRA)*, vol. 3, pp. 2724–2729 (1991)
7. Claes, P., Vandermeulen, D., Van Gool, L., Suetens, P.: Robust and accurate partial surface registration based on variational implicit surfaces for automatic 3D model building. In: *Fifth International Conference on 3-D Digital Imaging and Modeling (3DIM)* (2005)
8. CloudCompare: 3D point cloud and mesh processing software. <http://www.danielgm.net/cc/>. Accessed 9 Mar 2016
9. Curless, B., Levoy, M.: A volumetric method for building complex models from range images. In: *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1996*, pp. 303–312 (1996)
10. Dimashova, M., Lysenkov, I., Rabaud, V., Eruhimov, V.: Tabletop object scanning with an RGB-D sensor. In: *Third Workshop on Semantic Perception, Mapping and Exploration (SPME)* at the 2013 IEEE International Conference on Robotics and Automation (ICRA) (2013)
11. Handa, A., Whelan, T., McDonald, J., Davison, A.J.: A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. In: *IEEE International Conference on Robotics and Automation (ICRA)* (2014)
12. Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A., Fitzgibbon, A.: KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera. In: *ACM Symposium on User Interface Software and Technology (UIST)* (2011)
13. Kehl, W., Navab, N., Ilic, S.: Coloured signed distance fields for full 3D object reconstruction. In: *Proceedings of the British Machine Vision Conference (BMVC)* (2014)
14. Kerl, C.: GitHub - tum-vision/dvo: dense visual odometry. <https://github.com/tum-vision/dvo>. Accessed 9 Mar 2016
15. Kerl, C., Sturm, J., Cremers, D.: Robust odometry estimation for RGB-D cameras. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (2013)
16. Kubacki, D.B., Bui, H.Q., Babacan, S.D., Do, M.N.: Registration and integration of multiple depth images using signed distance function. In: *SPIE Proceedings*, vol. 8296 (2012)
17. Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K., Burgard, W.: g<sup>2</sup>o: a general framework for graph optimization. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3607–3613, May 2011

18. Lai, K., Bo, L., Ren, X., Fox, D.: A large-scale hierarchical multi-view RGB-D object dataset. In: IEEE International Conference on Robotics and Automation (ICRA) (2011)
19. Lorensen, W.E., Cline, H.E.: Marching cubes: a high resolution 3D surface construction algorithm. In: Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1987, pp. 163–169 (1987)
20. Lucas, B., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proceedings of the 7th International Joint Conference on Artificial Intelligence, IJCAI 1981, vol. 2, pp. 674–679 (1981)
21. Ma, Y., Soatto, S., Košecká, J., Sastry, S.S.: An Invitation to 3-D Vision: From Images to Geometric Models. Springer, New York (2003)
22. Masuda, T.: Registration and integration of multiple range images by matching signed distance fields for object shape modeling. *Comput. Vis. Image Underst. (CVIU)* **87**(1–3), 51–65 (2002)
23. Narayan, K.S., Sha, J., Singh, A., Abbeel, P.: Range sensor and silhouette fusion for high-quality 3D scanning. In: IEEE International Conference on Robotics and Automation (ICRA) (2015)
24. Newcombe, R.A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A.J., Kohli, P., Shotton, J., Hodges, S., Fitzgibbon, A.: KinectFusion: real-time dense surface mapping and tracking. In: 10th International Symposium on Mixed and Augmented Reality (ISMAR) (2011)
25. Osher, S., Fedkiw, R.: Level Set Methods and Dynamic Implicit Surfaces. Applied Mathematical Science, vol. 153. Springer, New York (2003)
26. Paragios, N., Rousson, M., Ramesh, V.: Non-rigid registration using distance functions. *Comput. Vis. Image Underst.* **89**(2–3), 142–165 (2003)
27. PCL: Point cloud library. <http://pointclouds.org/>. Accessed 9 Mar 2016
28. Pomerleau, F., Colas, F., Siegwart, R., Magnenat, S.: Comparing ICP variants on real-world data sets - open-source library and experimental protocol. *Auton. Robots* **34**(3), 133–148 (2013)
29. Pomerleau, F., Liu, M., Colas, F., Siegwart, R.: Challenging data sets for point cloud registration algorithms. *Int. J. Robot. Res. (IJRR)* **31**(14), 1705–1711 (2012)
30. Rouhani, M., Sappa, A.D.: The richer representation the better registration. *IEEE Trans. Image Proc.* **22**(12), 5036–5049 (2013)
31. Rusinkiewicz, S., Levoy, M.: Efficient variants of the ICP algorithm. In: 3rd International Conference on 3D Digital Imaging and Modeling (3DIM) (2001)
32. Rusu, R.B., Holzbach, A., Blodow, N., Beetz, M.: Fast geometric point labeling using conditional random fields. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2009)
33. Segal, A., Haehnel, D., Thrun, S.: Generalized-ICP. In: Proceedings of Robotics: Science and Systems (RSS) (2009)
34. Singh, A., Sha, J., Narayan, K., Achim, T., Abbeel, P.: BigBIRD: a large-scale 3D database of object instances. In: IEEE International Conference on Robotics and Automation (ICRA) (2014)
35. Stanford University: The Stanford 3D scanning repository. <http://graphics.stanford.edu/data/3Dscanrep/>. Accessed 9 Mar 2016
36. Steinbrücker, F., Sturm, J., Cremers, D.: Real-time visual odometry from dense RGB-D images. In: 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops) (2011)
37. Stoyanov, T., Magnusson, M., Lilienthal, A.: Point set registration through minimization of the  $L_2$  distance between 3D-NDT models. In: IEEE International Conference on Robotics and Automation (ICRA) (2012)



38. Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D.: A benchmark for the evaluation of RGB-D SLAM systems. In: Proceedings of the International Conference on Intelligent Robot Systems (IROS) (2012)
39. Tomasi, C., Manduchi, R.: Bilateral filtering for Gray and color images. In: Sixth IEEE International Conference on Computer Vision (ICCV), pp. 839–846 (1998)
40. Tubic, D., Hébert, P., Laurendeau, D.: A volumetric approach for interactive 3D modeling. *Comput. Vis. Image Underst. (CVIU)* **92**(1), 56–77 (2003)
41. Vijayanagar, K.R., Loghman, M., Kim, J.: Real-time refinement of Kinect depth maps using multi-resolution anisotropic diffusion. *Mobile Netw. Appl.* **19**(3), 414–425 (2014)
42. Zach, C., Pock, T., Bischof, H.: A globally optimal algorithm for robust TV- $L^1$  range image integration. In: Proceedings of the 11th IEEE International Conference on Computer Vision (ICCV), pp. 1–8 (2007)
43. ZCorporation: ZPrinter 650. Hardware manual (2008)
44. Zhou, Q., Koltun, V.: Dense scene reconstruction with points of interest. *ACM Trans. Graph.* **32**(4), 112:1–112:8 (2013)