3D Object Reconstruction from Hand-Object Interactions

Dimitrios Tzionas[1,2], Juergen Gall[1]
[1]Computer Vision Group, University of Bonn. [2]Max Planck Institute for Intelligent Systems.

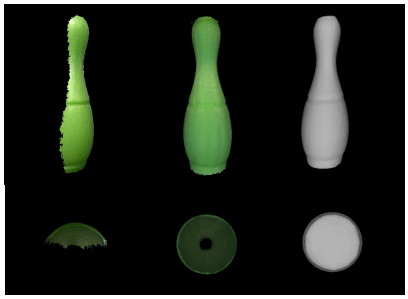

Figure 1: Reconstruction of a symmetric, textureless object (front and bottom view). Left: Existing in-hand scanning approaches fail for such objects. Middle and right: Successful reconstruction by the proposed in-hand scanning system that incorporates 3d hand motion capture.

Figure 2: The hand tracker used in the in-hand scanning pipeline. The left image shows the depth input map, the middle image shows the hand pose overlaid on the RGB-D data, while the right image shows just the hand pose.

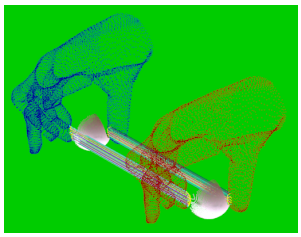

Figure 3: Contact correspondences $(X_{hand}, X'_{hand}) \in \mathcal{C}_{hand}(\theta, D_h)$ between the *source frame* (red) and the *target frame* (blue). The white point cloud is a partial view of the object to be reconstructed during hand-object interaction.

Recent advances have enabled a plethora of 3d object reconstruction approaches[1] using a single off-the-shelf RGB-D camera. Although these approaches are successful for a wide range of object classes, they rely on stable and distinctive geometric or texture features. Many objects like mechanical parts, toys, household or decorative articles, however, are textureless and characterized by minimalistic shapes that are simple and symmetric. Existing in-hand scanning systems and 3d reconstruction techniques fail for such symmetric objects in the absence of highly distinctive features. In this work we show that 3d reconstruction based on low-level features can be facilitated by higher level ones. Although existing in-hand scanning systems like [1, 4] simply ignore information originating from the hand, we show that 3D hand motion capture can provide strong and reliable features, effectively facilitating the reconstruction of even featureless, highly symmetrical objects, as the one shown in Figure 1.

To this end, we observe an RGB-D video where a hand is interacting with an object as illustrated in Figure 2. We track the hand pose and use the captured hand motion together with the object's texture and geometric features for object reconstruction as in Figure 3.

We first remove irrelevant parts of the RGB-D image D by thresholding the depth values, keeping only points within a specified volume. Subsequently we apply skin color segmentation on the RGB image using the Gaussian-Mixtures-Model (GMM) and get the masked RGB-D images D_o for the object and D_h for the hand.

In order to capture the motion of a hand, we employ an approach similar to [3]. The approach uses a hand template mesh and parameterizes the hand pose by a skeleton and linear blend skinning. For pose estimation, we minimize an objective function, which consists of three terms:

$$E(\theta, D) = \begin{aligned} & E_{model \to data}(\theta, D_h) + \\ & E_{data \to model}(\theta, D_h) + \gamma_c E_{collision}(\theta) \end{aligned} \quad (1)$$

where D_h is the current preprocessed depth image for the hand and θ are the pose parameters of the hand. The first two terms of Equation (1) minimize the alignment error between the input depth data and the hand pose. The alignment error is measured by $E_{model \to data}$, which measures how well the model fits the observed depth data, and $E_{data \to model}$, which measures how well the depth data is explained by the model. $E_{collision}$ penalizes finger intersections and ensures realistic, physically plausible poses.

In order to use the captured hand motion for 3D reconstruction, we have to infer the *contact points* with the object. For this we use the high-resolution mesh of the hand that is used for hand motion capture. To this end, we compute for each vertex associated to each end-effector the distance to the closest point of the object point cloud D_o. We first count for each end-effector the number of vertices with a closest distance of less than $1mm$. If an end-effector has more than 40 *candidate contact vertices*, it is labeled as a contact bone and all vertices of the bone are labeled as *contact vertices*. If there are not at least 2 end-effectors selected, we iteratively relax the distance threshold until we have at least two end-effectors. As a result, we obtain for each frame pair the set of *contact correspondences* $(X_{hand}, X'_{hand}) \in \mathcal{C}_{hand}(\theta, D_h)$, where (X_{hand}, X'_{hand}) is a pair of *contact vertices* in the *source* and *target* frame, respectively. Figure 3 depicts the *contact correspondences* for a frame pair.

For pairwise registration, we combine features extracted from D_o and the *contact points*, which have been extracted from D_h and the hand pose θ. As a result, we minimize an energy function based on two weighted energies:

$$E(\theta, D_h, D_o, \mathbf{R}, \mathbf{t}) = \begin{aligned} & E_{visual}(D_o, \mathbf{R}, \mathbf{t}) + \\ & \gamma_t E_{contact}(\theta, D_h, \mathbf{R}, \mathbf{t}) \end{aligned} \quad (2)$$

where E is a measure of the discrepancy between the incoming and the already processed data, that needs to be minimized. In that respect, we seek the rigid transformation $T = (\mathbf{R}, \mathbf{t})$, where $\mathbf{R} \in SO(3)$ is a rotation matrix and $\mathbf{t} \in \mathbb{R}^3$ is a translation vector, that minimizes the energy E by transforming the *source frame* accordingly.

The *visual energy* E_{visual} consists of two terms that are computed on the visual data of the object point cloud D_o:

$$E_{visual}(D_o, \mathbf{R}, \mathbf{t}) = \begin{aligned} & E_{feat2d}(D_o, \mathbf{R}, \mathbf{t}) + \\ & E_{feat3d}(D_o, \mathbf{R}, \mathbf{t}) \end{aligned} \quad (3)$$

The term E_{feat2d} is based on a sparse set of correspondences $\mathcal{C}_{feat2d}(D_o)$ using $2d$ SIFT features that are back-projected in $3d$ by the function $\varphi(x)$: $\mathbb{R}^2 \to \mathbb{R}^3$, given the intrinsic parameters of the camera. The $2d$ SIFT keypoint correspondences in the source and target image respectively are denoted as $(x_{2d}, x'_{2d}) \in \mathcal{C}_{feat2d}(D_o)$, while $X_{2d} = \varphi(x_{2d})$ and $X'_{2d} = \varphi(x'_{2d})$ are the corresponding back-projected $3d$ points. E_{feat2d} is then formulated

[1]e.g. Kinect-Fusion, Shapify, Fablitec, Skanect, iSense, KScan3d

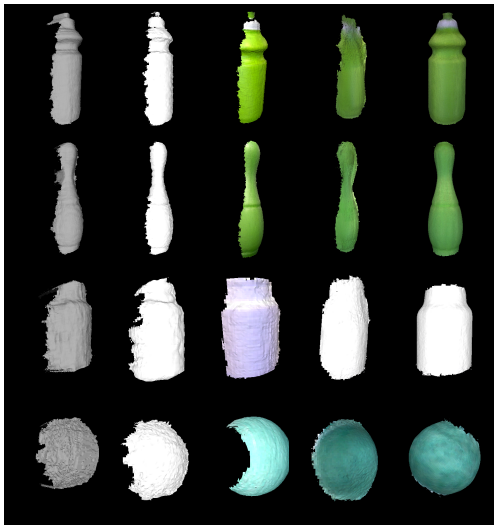

Figure 4: Qualitative comparison of different in-hand scanning systems for our four objects. We visualize the meshes extracted from the TSDF volume. From left to right, each row contains the result of: (a) KinFu, (b) Skanect, (c) Our pipeline with a turntable and without hand motion data, (d) Our pipeline with in-hand scanning but without hand motion data, (e) Our pipeline with in-hand scanning that includes hand motion data (the proposed setup). Only the last one succeeds in reconstructing all symmetric objects.

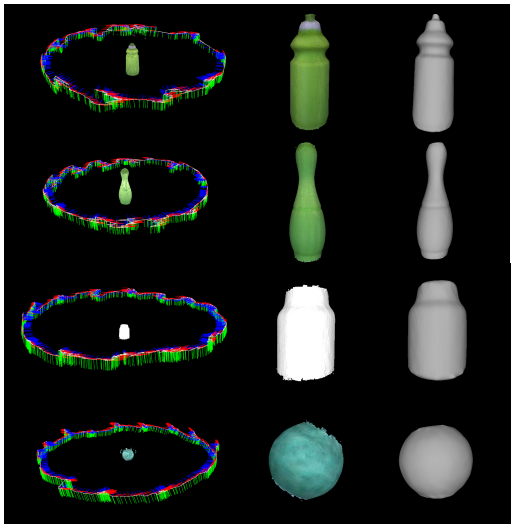

Figure 5: Qualitative results of our pipeline for our four objects when a hand rotates the object in front of the camera. The left images show the reconstructed camera poses. The poses follow a circular path, whose shape signifies the type of hand-object interaction during the rotation. The middle images show the mesh that is acquired with marching cubes from the TSDF volume, while the right ones show the final water-tight mesh that is acquired with Poisson reconstruction.

as

$$E_{feat2d}(D_o,\mathbf{R},\mathbf{t}) = \sum_{(X_{2d},X'_{2d})\in\mathcal{C}_{feat2d}} \|X'_{2d} - (\mathbf{R}X_{2d}+\mathbf{t})\|^2. \quad (4)$$

In a similar manner, the term E_{feat3d} is based on a sparse set of correspondences $\mathcal{C}_{feat3d}(D_o)$. Instead of the image domain, we operate on the $3d$ point cloud by choosing ISS3D keypoints and the CSHOT feature descriptor. E_{feat3d} is then formulated as

$$E_{feat3d}(D_o,\mathbf{R},\mathbf{t}) = \sum_{(X_{3d},X'_{3d})\in\mathcal{C}_{feat3d}} \|X'_{3d} - (\mathbf{R}X_{3d}+\mathbf{t})\|^2. \quad (5)$$

Finally, the term $E_{contact}$ depends on the current hand pose estimate θ and the hand point cloud D_h. Based on which the *contact correspondences* are computed as described above. Let $(X_{hand},X'_{hand}) \in \mathcal{C}_{hand}(\theta,D_h)$ be the corresponding *contact points*, i.e. vertices, in the *source* and *target* frame respectively, then $E_{contact}(\theta,D_h)$ is written as

$$E_{contact}(\theta,D_h,\mathbf{R},\mathbf{t}) = \sum_{(X_{hand},X'_{hand})\in\mathcal{C}_{hand}} \|X'_{hand} - (\mathbf{R}X_{hand}+\mathbf{t})\|^2. \quad (6)$$

The two terms in the energy function (2) are weighted since they have different characteristics. Although *visual correspondences* preserve local geometric or textural details better, they tend to cause a slipping of one frame upon another in case of textureless and symmetric objects. In this case, the *contact correspondences* ensure that the movement of the hand is taken into account.

The sparse correspondence sets $\mathcal{C}_{feat2d}$, $\mathcal{C}_{feat3d}$, and $\mathcal{C}_{hand}$ provide usually an imperfect alignment of the *source* frame to the *target* frame either due to noise or ambiguities in the visual features or the pose. For this reason, we refine the aligned source frame by finding a locally optimal solution based on dense ICP correspondences. During this refinement stage we align the current frame to the accumulation of all previously aligned frames, i.e. the current partial reconstructed model. After finding a dense set $(X_{icp},X'_{icp}) \in \mathcal{C}_{icp}(D_o)$ of ICP correspondences with maximum distance of 5mm, we minimize the discrepancy between them

$$E_{icp}(D_o,\mathbf{R},\mathbf{t}) = \sum_{(X_{icp},X'_{icp})\in\mathcal{C}_{icp}} \|X'_{icp} - (\mathbf{R}X_{icp}+\mathbf{t})\|^2. \quad (7)$$

After aligning all the frames, we need a mesh representation of the reconstructed object. To this end we first employ a TSDF volume to get a volumetric representation. Subsequently we apply the marching-cubes method to extract a mesh and remove tiny disconnected components. The final mesh is then obtained by Laplacian smoothing followed by Poisson reconstruction to get a smooth, water-tight mesh with preserved details.

For evaluation in comparison to the state-of-the-art 3d reconstruction methods *KinFu*[2] and *Skanect*[3] we perform with them 3d reconstruction of our four objects. Figure 4 shows the results both with and without the use of hands and hand motion data. The images show that the reconstruction without hands is similar across different systems and results in a degenerate 3d representation of the object. The incorporation of hand motion capture in the reconstruction plays clearly a vital role, leading to the effective reconstruction of the full surface of the object.

Although Figure 4 compares the TSDF meshes, more detailed results are shown in Figure 5. The camera poses are reconstructed effectively, showing not only the rotational movement during the scanning process, but also the type and intensity of hand-object interaction. The water-tight meshes that are shown compose the final output of our system. The resulting reconstruction renders our approach the first in-hand scanning system to cope with the reconstruction of symmetric objects, while also showing prospects of future practical applications.

Further details and experiments can be found in [2]. The recorded sequences, calibration data, hand motion data, as well as video results, the resulting meshes and the source code for reconstruction are publicly available at http://files.is.tue.mpg.de/dtzionas/In-Hand-Scanning.

Acknowledgement: Financial support was provided by the DFG Emmy Noether program (GA 1927/1-1).

[1] Szymon Rusinkiewicz, Olaf Hall-Holt, and Marc Levoy. Real-time 3d model acquisition. *TOG*, 21(3):438–446, 2002.

[2] Dimitrios Tzionas and Juergen Gall. 3d object reconstruction from hand-object interactions. In *ICCV*, 2015.

[3] Dimitrios Tzionas, Abhilash Srikantha, Pablo Aponte, and Juergen Gall. Capturing hand motion with an rgb-d sensor, fusing a generative model with salient points. In *GCPR*, 2014.

[4] Thibaut Weise, Thomas Wismer, Bastian Leibe, and Luc Van Gool. Online loop closure for real-time interactive 3d scanning. *CVIU*, 2011.

[2] http://pointclouds.org/documentation/tutorials/using_kinfu_large_scale.php

[3] http://skanect.occipital.com