RGB-D Object Modelling for Object Recognition and Tracking

Johann Prankl, Aitor Aldoma, Alexander Svejda and Markus Vincze

***Abstract*— This work presents a flexible system to reconstruct 3D models of objects captured with an RGB-D sensor. A major advantage of the method is that unlike other modelling tools, our reconstruction pipeline allows the user to acquire a full 3D model of the object. This is achieved by acquiring several partial 3D models in different sessions — each individual session presenting the object of interest in different configurations that reveal occluded parts of the object — that are automatically merged together to reconstruct a full 3D model. In addition, the 3D models acquired by our system can be directly used by state-of-the-art object instance recognition and object tracking modules, providing object-perception capabilities to complex applications requiring these functionalities (e.g. human-object interaction analysis, robot grasping, etc.). The system does not impose constraints in the appearance of objects (textured, untextured) nor in the modelling setup (moving camera with static object or turn-table setups with static camera). The proposed reconstruction system has been used to model a large number of objects resulting in metrically accurate and visually appealing 3D models.**

I. INTRODUCTION

The availability of commodity RGB-D sensors, combined with several advances in 3D printing technology, has sparked a renewed interest in software tools that enable users to digitize objects easily, and most importantly, at low economical costs. However, being able to accurately reconstruct 3D object models has not only applications among modelling or 3D printing aficionados, but also in the field of robotics. For instance, the information in form of 3D models can be used for object instance recognition, enabling applications such as autonomous grasping, or object search under clutter and occlusions.

While numerous reconstruction tools exist to capture 3D models of environments, only a few of them focus on the reconstruction of individual objects. This can be partially ascribed to the difference in scale between objects (e.g. household objects) and larger environments (e.g. rooms or buildings, usually the focus of SLAM systems), the need to subtract the object of interest from the rest of the environment, as well as other nuisances that make object reconstruction a challenging problem. For example, the requirement of full 3D models is ignored by most reconstruction systems.

Addressing the aforementioned challenges, we propose an integrated reconstruction pipeline in order to enable recognition and tracking of object. Our contributions are: (i) a novel approach which is able to reconstruct full 3D models by merging partial models acquired in different sessions and (ii) results in metrically accurate and visually appealing models, (iii) a system which is easy to use, (iv) does not make assumptions of the kind of objects being modelled[1] and (v) is able to export object models that can be seamlessly integrated into object recognition and tracking modules without any additional hassle. The latter being able to facilitate research in robotic areas that require 3D models or tracking and recognition capabilities. Therefore, we will release our modelling and object perception systems to enable this.

Fig. 1. Virtual scene recreated with some of the 3D models reconstructed by the proposed modelling tool.

In the remainder of this paper, we present the different modules of the system, focusing on those with novel characteristics or that are crucial to the robustness of the overall system. Because the evaluation of complex pipelines like the one proposed in this paper is always a major challenge, we compare the fidelity of the end result (i.e. 3D models) obtained with our system with their counterparts reconstructed using a precise laser scanner. This quantitative comparison shows that the reconstructed models are metri-

Johann Prankl, Aitor Aldoma, Alexander Svejda and Markus Vincze are with the Vision4Robotics group (ACIN - Vienna University of Technology), Austria {prankl, aldoma, vincze}@acin.tuwien.ac.at

[1] As long as they can be sensed by RGB-D sensors

cally accurate: the average error ranging between one and two millimetres. We also show how the reconstructed 3D models are effectively used for object instance recognition and 6-DoF pose estimation, as well as for object tracking with monocular cameras.

II. RELATED WORK

The proposed framework covers a broad variety of methods including registration, object segmentation, surface reconstruction, texturing, and supported applications such as object tracking and object recognition. In this section we focus on related work of the core methods necessary for object modelling: camera tracking, point cloud registration and surface modelling.

Since Lowe developed the Scale Invariant Feature Transform (SIFT) in 2004 [1], using interest points is the most popular way of finding correspondences in image pairs enabling the registration of RGB-D frames. For example Endres et al. [2] developed a Visual SLAM approach which is able to track the camera pose and register point clouds in large environments. Loop closing and a graph based optimization method are used to compensate for the error accumulated during camera tracking. Interest points can also be used to directly reconstruct models for object recognition. In [3] Collet et al. register a set of images and compute a spare recognition model using a Structure from Motion approach. Especially for re-localization we also rely on interest points. In contrast to Endres et al. [2] we develop a LK-style tracking approach which is able to minimize the drift, enabling the creation of models for tracking and recognition without the necessity of an explicit loop closing.

Another type of methods is based on the well established Iterative Closest Point (ICP) algorithm [4], [5], [6], [7]. Huber et al. [4] as well as Fantoni et al. [5] focus on the registration of unordered sets of range images, while Weise et al. [6] track range images and propose an online loop closing approach. In [7] the authors propose a robotic in-hand object modelling approach where the object and the robotic manipulator are tracked with an articulated ICP variant.

While the above systems generate sparse representations, namely point clouds, the celebrated approach of Izadi et al. [8] uses a truly dense representation based on signed distance functions [9]. Since then, several extensions of the original algorithm have appeared [10], [11]. While the original Kinect Fusion [8] relies on depth data Kehl et al. [10] introduce a colour term and is like our proposal able to register multiple modelling sessions. However, [10] relies on sampling the rotational part of the pose space in order to provide initial approximations to their registration method. Instead, we use features and stable planes to attain initial alignments effectively reducing computational complexity. A direct approach for registration is proposed in Bylow et al. [11]. They omit ICP and directly optimize the camera poses using the SDF-volume. Furthermore, the first commercial scanning solutions such as ReconstructMe, itSeez3D [12] and CopyMe3D [13] became available.

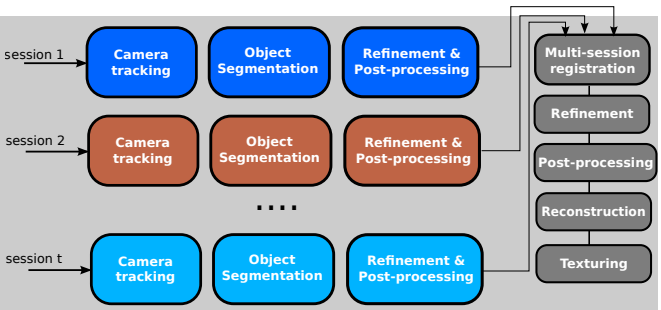

Fig. 2. Pictorial overview of the proposed object modelling pipeline.

In summary, we propose a robust and user-friendly approach which is flexible enough to adapt to different user requirements and is able to generate object models for tracking and recognition. Hence, the application of our framework does not primarily focus on augmented reality or commercial 3D printing applications, but especially on the object perception requirements of the robotics community to enable robots in household environments.

III. SYSTEM OVERVIEW

Approaches for object modelling typically involve accurate camera tracking, object segmentation and, depending on the application, a post-processing step which includes pose refinement and eventually surface reconstructing and texturing. Concerning camera tracking, we use a visual odometry based on tracked interest points. If the object itself is texture-less, we rely on background texture (e.g. by adding a textured sheet of paper on the supporting surface) in order to successfully model these kind of objects. The camera positions are refined by means of bundle adjustment as well as an accurate multi-view ICP approach.

Segmentation of the object of interest from the background is attained by a multi-plane detection and a smooth clustering approach offering object hypotheses to be selected by the user. Alternatively, a simple bounding box around the object can be used to define a region of interest from which the object is easily singled out from the background.

If a complete model (i.e. including the bottom and self-occluded parts) is desired, a registration approach is proposed to automatically align multiple sequences. Finally, our system includes a post-processing stage to reduce artefacts coming from noisy observations as well as a surface reconstruction and texturing module to generate dense and textured meshes. A schematic representation of the modelling pipeline is depicted in Figure 2. The individual steps including novel aspects of the system are explained in more detail in the following sections.

IV. REGISTRATION AND SEGMENTATION

A key component for the reconstruction of 3D models is the ability to accurately track the camera pose with respect to the object of interest. This section discusses the selected procedure for this task as well as the different

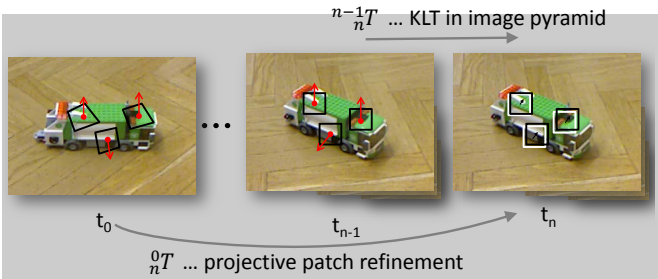

Fig. 3. Camera tracking including a frame by frame visual odometry and a projective patch refinement from keyframe to frame.

strategies available to single out the object of interest from the background.

A. Camera tracking and keyframe selection

The proposed approach combines frame by frame tracking based on a KLT-tracker [14] and a keyframe based refinement step by projecting patches to the current frame and optimizing their locations. To estimate the camera pose, the rigid transformation is computed from the corresponding depth information of the organized RGB-D frames. Fig. 3 depicts the tracking approach, where T indicates the pose transformations computed from tracked points.

In more detail, a keyframe is initialized by detecting FAST-keypoints [15] and assigning them to the corresponding 3D locations. The keypoints are then tracked frame by frame using a pyramidal implementation of the KLT-tracker, which allows to track fast camera motions with a reasonable amount of motion blur. Then the corresponding 3D points are used to robustly estimate the rigid transformation using RANSAC. To account for the accumulated drift as well as to compute a confidence value for individual point correspondences we developed a projective patch refinement. Therefore, once a keyframe is created, normals are estimated and in combination with the pose hypothesis a locally correct patch warping (homography) from the keyframe to the current frame is performed. An additional KLT-style refinement step including the normalized cross correlation of the patches gives a sub-pixel accurate location and a meaningful confidence value. This method is able to reduce the drift while tracking and provides sub-pixel accurate image locations for bundle adjustment.

Keyframes are generated depending on the tracked camera pose, hence our framework is not only able to model table top objects, but also larger environments. During tracking previously visited camera locations are tested and if there is a known view point – i.e., the difference of the current camera location to that of a stored keyframe is within a threshold – the system tracks that keyframe instead of generating a new one. This avoids storing redundant information and these loops are further used to improve the camera locations in a post-processing step using bundle adjustment. Note, for a spatially constrained environment and by using our high accurate camera tracking algorithm it is not necessary to integrate more sophisticated loop closing algorithms. In addition once the camera tracker fails and poses get uncertain we use the keypoint descriptor proposed in [16] for re-localization.

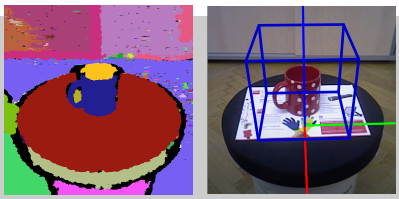

Fig. 4. Labels of planes and smooth clusters (left) used for automatic adjustment of region of interests (right) and for interactive object segmentation.

This stage results in a set of keyframes $\mathcal{K} = \{K^1, ..., K^n\}$ and a set of transformations $\mathcal{T} = \{T^1, ..., T^n\}$ aligning the corresponding keyframes to the reference frame of the reconstructed model. The reference frame is either defined by the first camera frame or by a user defined region of interest (cf. next section).

B. Object-background segmentation

The camera tracking framework described in the previous section is already capable of modelling complete scenes in real-time. If one wants to reconstruct individual objects an additional manual interaction is necessary. We provide two options to segment objects, namely

- an interactive segmentation approach, and
- segmentation based on a tracked region of interest.

In the optimal case both variants are able to segment objects with a single mouse click. The interactive segmentation relies on multi-plane detection and smooth segmentation (Fig. 4, left). Flat parts, larger than a certain threshold are modelled as planes and the remaining areas are recursively clustered depending on the deviation of the surface normals of neighbouring image points. Hence, smooth clusters “pop out” from the surrounding planar surfaces and need to be selected to form up a complete object.

The second option we implemented is to select a planar surface before the camera tracking starts. This automatically computes a region of interest (ROI) around the surface, which is used to constrain the feature locations used for camera tracking and to segment the object above the plane in a post-processing step (Fig. 4, right). Hence, a single click suffices and the whole modelling process is performed automatically. This method can also be used to model an object on a turn-table because the surrounding static environment is not considered for tracking.

The result of this stage is a set of indices $\mathcal{I} = \{I^1, ..., I^{n_i}\}$, I^k indicating the pixels of K^k containing the object of interest. An initial point cloud of the object can be reconstructed as $\mathcal{P} = \bigcup_{k=1:n} T^k \left(K^k[I^k]\right)$ where $K[\cdot]$ indicates the extraction of a set of indices from a keyframe.

C. Multi-view refinement

While the visual odometry presented in Section IV-A has proven to be sufficiently accurate for the envisioned scenario, the concatenation of several transformations inevitably results in some amount of drift in the overall registration. Aiming at mitigating this undesirable effect as well as in order to take advantage of the multiple observations with significant overlap, our framework is equipped with two alternative mechanisms to reduce the global registration error.

On one hand, the framework allows to perform bundle-adjustment in order to reduce the re-projection error of correspondences used during camera tracking. On the other hand, the system is equipped with the multi-view Iterative Closest Point introduced in [17] that globally reduces the registration error between overlapping views by iteratively adapting the transformation between camera poses. While multi-view ICP is considerable slower than bundle-adjustment, its application is not constrained to objects with visual features and due to its dense nature, results in more accurate registrations.

Both processes update the transformation set $\mathcal{T}$ introduced in previous sections.

V. Post-processing

The methods presented so far have been designed to be robust to noise and sensor nuisances. However, such artefacts are present in the data and a post-processing stage is required to remove them in order to obtain a visually appealing and accurate model. The techniques within this section provide a improved reconstruction by removing these artefacts from the underlying data. Figure 5 visualizes the improvement on the final reconstruction after the post-processing stage. Please note that the methods herein, do not change the alignment results obtained during the registration process.

Fig. 5. Effects of the post-processing stage on the reconstruction results.

A. Noise model

In [18], the authors study the effect of surface-sensor *distance* and *angle* on the data. They obtain axial and lateral noise distributions by varying the aforementioned two variables and show how to include the derived noise model into Kinect Fusion [8] to better accommodate noisy observations in order to reconstruct thin and challenging areas.

In particular, for object modelling, *surface-sensor angle* is more important than distance, since the later can be controlled and kept at an optimal range (i.e., one meter or closer). Following [18], we observe that:

- Data quickly deteriorates when the angle between the sensor and the surface gets above 60 degrees.
- Lateral noise increases linearly with distance to the sensor. It results in jagged edges close to depth discontinuities causing the measured point to jump between foreground and background. Combining depth with colour information makes this effect clearly visible as colour information from the background appears on the foreground object and vice-versa. Observe the white points on the left instances of reconstructed models in Figure 5 coming from the plane on the background where the objects are standing.

From the previous two observations, we propose a simple noise model suited for object modelling that results in a significant improvement on the visual quality of the reconstruction. Let $\mathcal{C} = \{p_i\}$ represent a point cloud in the sensor reference frame, $\mathcal{N} = \{n_i\}$ the associated normal information and $\mathcal{E} = \{e_i\}$, e_i being a boolean variable indicating whether p_i is located at a depth discontinuity or not. w_i is readily computed as follows:

$$w_i = \left(1 - \frac{\theta - \theta_{max}}{90 - \theta_{max}}\right) \cdot \left(1 - \frac{1}{2} exp^{-\frac{d_i^2}{\sigma_L^2}}\right) \qquad (1)$$

where θ represents the angle between n_i and the sensor, $\theta_{max} = 60°$, $d_i = ||p_i - p_j||_2$ (p_j being the closest point with $e_j = true$) and $\sigma_L = 0.002$ represents the lateral noise sigma.

B. Exploiting noise model and data redundancy

Because the selected keyframes present a certain overlap, we improve the final point cloud by averaging good (based on the noise model weights) observations that lie on the same actual surface as well as by removing inconsistent observations. To do so, we iterate over all keyframes and for each keyframe, K, project the points $p \in \mathcal{P}$ into $(u, v) \in K^2$. If the point and its projection are inconsistent (i.e. they do not lie on the same surface), we mark the point p as invalid if its associated noise weight is smaller than the weight associated with the projection $(u, v) \in K$.

The previous step effectively removes inconsistent observations from the object reconstruction. Finally, the remaining observations are averaged together by putting all points into an octree structure with a certain leaf resolution[3]. A representative for each leaf is computed from all points falling within the leaf boundaries by means of a weighted average (weights coming again from the noise model).

VI. Multi-session alignment

In this section, we discuss the proposed techniques to automatically align multiple sessions into a consistent 3D model. Please note that since the configuration of the object has been changed with respect to its surroundings (e.g.

[2]This is attained by means of the inverse transformation aligning the different keyframes into the reference frame of the model combined with the projection matrix of the sensor.

[3]We use a resolution of 1mm for our experiments.

supporting plane), this process needs to rely solely on the information provided by the object. Figure 6 shows an object in three different sessions as well as the reconstructed point cloud.

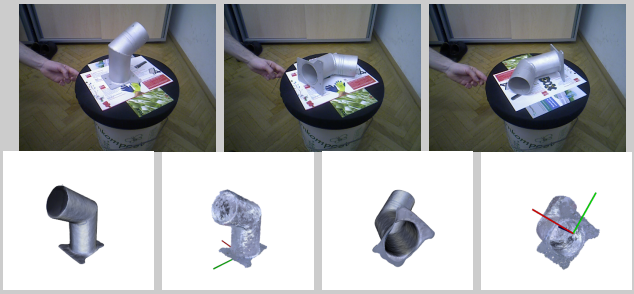

Fig. 6. *Top row*: Pipe object in different configurations (three sessions). *Bottom row*: textured poisson reconstruction and reconstructed point cloud.

Let $\mathcal{P}_{1:t}$ be a set of t partial 3D models obtained by reconstructing the same object in different configurations. The goal now is to find a set of transformations that align the different scans, $\mathcal{P}_{1:k}$, into the coordinate system of (without loss of generality) $\mathcal{P}_1$. For simplicity, let us discuss first the case where $t = 2$.

In this case, we seek a single transformation aligning $\mathcal{P}_2$ to $\mathcal{P}_1$. To obtain it, we make use of the initial alignments provided by the methods discussed later on in Section VI-A. Each initial alignment is then refined by means of ICP. Because several initial alignments can be provided, we need to define a metric to evaluate the registration quality. The transformation associated with the best registration according to this criteria will be then the sought transformation. This quality criterion is based on two aspects: (i) number of points causing free space violation (FSV) and (ii) amount of overlap. Recall from [19] that the FSV ratio between two point clouds is efficiently computed as the ratio of the number of points of the first cloud in front of the surface of the second cloud over the number of points in the same surface. Intuitively, we would like on one hand to favour transformations causing a small number of free space violations (indicating consistent alignments) and on the other hand, to favour alignments that present enough overlap to compute an accurate transformation.

If $t \geq 3$, we repeat the process above for all pairs $(\mathcal{P}_i,\mathcal{P}_j)_{i>j}$. Then, we create a weighted graph with k vertices and edges between vertices including the best transformation aligning $(\mathcal{P}_i,\mathcal{P}_j)$ together with the computed quality measure. Then, a unique registration of all partial models is obtained by computing the MST of the graph and appropriately concatenating the transformations found at the edges of the tree when traversing from $\mathcal{P}_i$ to $\mathcal{P}_1$. After all partial models have been brought into alignment, the multi-view refinement process as well as the post-processing stage previously described may be executed for further accuracy.

A. Initial alignment of multiple sessions

This section discusses two complementary alternatives to provide the initial alignments between pairs of sessions. The first one is based on appearance and/or geometrical features on the object that can be matched across different sessions. The second technique is based on the fact that objects are modelled on a supporting surface, thus constraining the possible configurations of the object on the supporting surface to configurations on which the object remains stationary. This intuition is exploited to reduce the degrees of freedom when estimating transformations between two sessions of the same object.

1) Feature-based registration: If a pair of sessions present enough common features (at least 3), it is possible to estimate the rigid transformation aligning two partial models. Correspondences between bodies are obtained by matching SIFT [1] and SHOT [20] features (capturing thus both appearance and geometrical information). Resiliency to outliers is attained, as commonly done in object recognition pipelines, by deploying a correspondence grouping stage followed by RANSAC and absolute orientation estimation. Because of the correspondence grouping stage, several transformations are estimated, representing the initial alignments fed into the previous algorithm. More details of similar techniques used in local recognition pipelines can be found in [21].

2) Stable planes registration: Alternatively, a complementary set of initial alignments can be obtained by using the modelling constraint that objects lie on a planar surface. Therefore, the stable planes of $\mathcal{P}_i$ are used to bootstrap initial alignments between $\mathcal{P}_i$ and $\mathcal{P}_j$. Intuitively, one of the stable planes of $\mathcal{P}_i$ might be the supporting surface on which $\mathcal{P}_j$ is modelled. As described in [22], stable planes can be efficiently computed by merging the faces of the convex hull with similar normals. Please note, that aligning planes locks 3 of the 6 degrees of freedom involved in rigid body registration. The remaining 3 degrees of freedom (i.e. translation on the plane and rotation about the plane normal) are respectively approximated by centring the point cloud and by sampling rotations about the plane normal (every 30° in our settings). To speed up the computation of initial alignments, only the 4 most probable [4] stable planes of $\mathcal{P}_i$ are used. This combination results in 48 initial alignments that are refined by means of ICP. Figure 7 shows two examples where the objects do not have enough features to be matched across sessions (due to repetitive structure) that are however correctly aligned using stable planes.

VII. Surface reconstruction and texturing

In order to extract a dense surface from the reconstructed point cloud, we rely on Poisson Surface Reconstruction [23]. The method finds a globally consistent surface that fits the sparse data accurately avoiding over-smoothing or over-fitting. A polygonal mesh is then extracted by an adapted version of the Marching Cubes algorithm [24]. One problem of Poisson Reconstruction, which is also mentioned in the original paper, occurs when the algorithm is applied to point clouds containing holes (i.e. parts of the objects where not

[4]Based on the total area of the supporting faces of the convex hull.

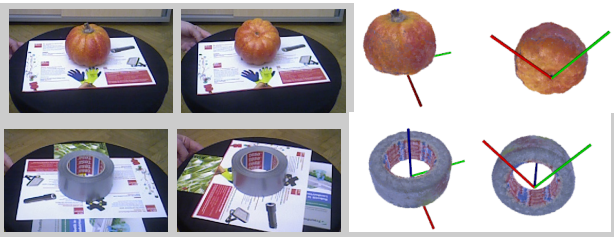

Fig. 7. Examples of successful alignments between sessions by means of stable planes. The objects do not present enough unique features matchable across sessions to enable registration using features.

seen). This is usually the case when dealing with objects reconstructed from a single sequence, where the bottom of the model is not defined. In these cases, poisson reconstruction tends to add an extension to the reconstructed surface as shown on the left hand side of Figure 8. To overcome this, we first estimate the convex hull of the point cloud. Next, all vertices of the mesh that lie outside the convex hull are projected onto the surface of the hull, thus ensuring that no mesh vertices lie outside. The right hand side of Figure 8 shows the resulting mesh.

Fig. 8. Reconstructed polygon mesh before and after cropping using the convex hull of the reconstructed point cloud.

To texture the model, we use the multi-band blending approach proposed in [25]. In a nutshell, the texturing algorithm consists of two steps. First, each face of the reconstructed mesh is mapped to one of the input views. To avoid highly fragmented textures, only a subset of the views is taken into account. This subset is obtained by defining candidate views for each face based on the angle between the face normal and the view ray of the camera. Then, the minimal set of views is selected such that all mesh faces are covered.

The second step aims at improving visual quality of the resulting texture map. Due to inaccurate camera calibration and small registration errors, the texture at boundaries between two views might show artefacts such as incorrect positioning and colour inconsistency. In order to achieve smooth transitions between texture patches, a multi-band blending technique is applied. First, each view is decomposed into different frequency components using Laplacian pyramids, which are approximated through difference of Gaussian pyramids. Finally, each pixel of the texture map is blended from multiple views based on the viewing angle: Higher frequency parts are blended only from views with small viewing angle, whereas lower frequency parts of the image are blended from a broader viewing range. This method allows smooth blending and preservation of texture details without introducing ghosting artefacts.

VIII. EXPERIMENTAL RESULTS

In addition to the qualitative results shown throughout this work, this section evaluates (i) the accuracy of the reconstructed models with respect to models of the same objects acquired with a precise Laser Scanner [26] and (ii) if the reconstructed models are accurate enough for the tasks of object instance recognition and pose estimation as well as object tracking from monocular cameras, the latter being one of the main goals of this work in order to facilitate the usage of object perception systems previously developed.

A. Comparison with Laser Scanner models

We assess quantitatively the quality of our reconstructions by comparing the reconstructed 3D models with their counterparts from the KIT Object Models Web Database [26]. To do so, we use the CloudCompare software [5] in order to interactively register both instances of the objects and to compute quality metrics. In particular, the error is assessed by computing statistics regarding the closest distance from the reconstructed point cloud to the mesh provided by [26]. Figures 9 to 11 show the computed statistics on three objects. The average error as well as the standard deviation indicate that the quality of the models lies within the noise range of the sensor at the modelling distance. Moreover, the error distributions are comparable to those reported by [10] that uses a similar evaluation metric.

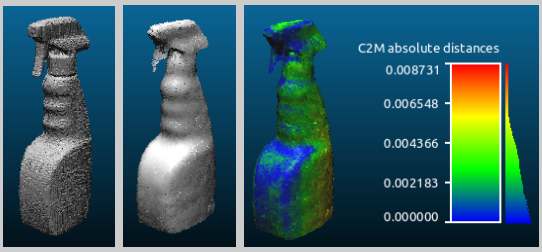

Fig. 9. Distance from reconstructed point clouds (middle) against laser scanner model (left). Distance ($\mu \pm \sigma$): 2.16 $\pm$ 1.53mm

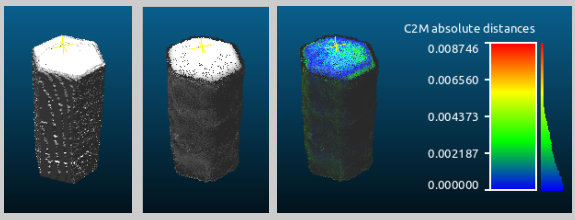

Fig. 10. Distance from reconstructed point clouds (middle) against laser scanner model (left). Distance ($\mu \pm \sigma$): 1.82 $\pm$ 1.44mm

[5] http://www.danielgm.net/cc/

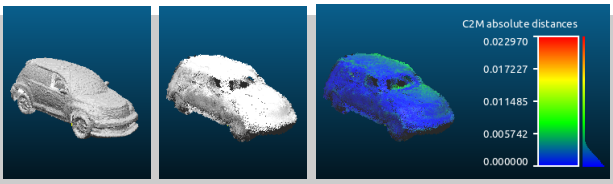

Fig. 11. Distance from reconstructed point clouds (middle) against laser scanner model (left). Distance ($\mu \pm \sigma$): 1.71 $\pm$ 1.96mm

B. Object recognition

The 3D models reconstructed with the proposed pipeline as well as the information gathered during the modelling process are of great value for object instance recognition. For this reason, the modelling tool enables users to export object models in the format required by the recognition pipeline proposed in [27]. In particular, the selected keyframes, object indices as well as camera poses are exported together with the reconstructed 3D point cloud of the object. While the 3D point cloud of the objects is used in the hypothesis verification process of [27], the individual keyframes are used to learn features that allow to find correspondences between the scene and the object models. Figure 12 shows the recognition results obtained by an improved version of [27] on a complex scene. The quality of the recognition results indicates that the reconstructed models are accurate enough for the recognition task, implicitly validating the reconstruction pipeline proposed in this work.

C. Object tracking

The camera tracking approach described in Section IV-A is based on interest points for initialization and LK tracking. This results in a model which can directly be used for object tracking. Hence, we integrated an export function to segment the interest points and store them including the images of the keyframes. To test the object tracking we use a method similar to the camera tracking approach, but instead of computing the rigid transformation base on RGB-D images we estimate the pose with a pnp-algorithm. Thus the object tracker is able to track the 6-DoF pose from a monocular image sequence. Figure 13 shows examples of the sparse interest point model, the tracked trajectories and selected frames where the object is near the camera and frames at the maximum tracked distance. The upper row depicts a successful tracking result with a maximum distance of 2m to the camera. A more challenging example is shown in the second row where a rather small object gets lost at a distance of about 1.15m.

IX. CONCLUSIONS

In this paper we have presented a flexible object reconstruction pipeline. Unlike most of the reconstruction and modelling tools out there, our proposal is able to reconstruct full 3D models of objects by changing the object configuration across different sessions. We have shown how the registration of different sessions can be carried out on featureless objects by exploiting the modelling setup where objects lie on a stable surface. Another key functionality of our proposal is the ability to export object models in such a way that they can directly be used for object recognition and tracking. With this respect the proposed framework supersedes the publicly available toolbox BLORT [28], where object modelling is a somewhat tedious process. We believe that these tools will facilitate research in areas requiring object perception (e.g. human-object or robot-object interaction, grasping, object search as well as planning systems).

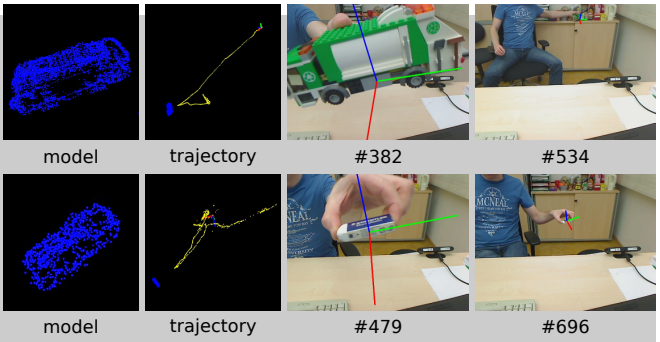

Fig. 13. Examples of tracked objects with the interest points model (left), the tracked trajectory 2^{nd} column, the nearest frame and the frame with the largest distance to the camera.

ACKNOWLEDGMENT

The research leading to these results has received funding from the European Community Seventh Framework Programme FP7/2007-2013 under grant agreement No. 600623, STRANDS, No. 288146, HOBBIT and No. 610532, SQUIRREL and by Sparkling Science – a programme of the Federal Ministry of Science and Research of Austria (SPA 04/84, FRANC).

REFERENCES

[1] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91–110, 2004.

[2] F. Endres, J. Hess, J. Sturm, D. Cremers, and W. Burgard, "3-d mapping with an rgb-d camera," *Robotics, IEEE Transactions on*, pp. 177–187, 2014.

[3] A. Collet Romea, D. Berenson, S. Srinivasa, and D. Ferguson , "Object recognition and full pose registration from a single image for robotic manipulation," in *IEEE International Conference on Robotics and Automation (ICRA '09)*, May 2009.

[4] D. F. Huber and M. Hebert, "Fully automatic registration of multiple 3d data sets," *Image and Vision Computing*, vol. 21, no. 7, pp. 637 – 650, 2003, computer Vision beyond the visible spectrum.

[5] S. Fantoni, U. Castellani, and A. Fusiello, "Accurate and automatic alignment of range surfaces," in *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference on*, Oct 2012, pp. 73–80.

[6] T. Weise, T. Wismer, B. Leibe, and L. V. Gool, "Online loop closure for real-time interactive 3d scanning," *Computer Vision and Image Understanding*, vol. 115, no. 5, pp. 635 – 648, 2011, special issue on 3D Imaging and Modelling.

[7] M. Krainin, P. Henry, X. Ren, and D. Fox, "Manipulator and object tracking for in-hand 3d object modeling," *Intl Journal of Robotics Research (IJRR*, pp. 1311–1327, 2011.

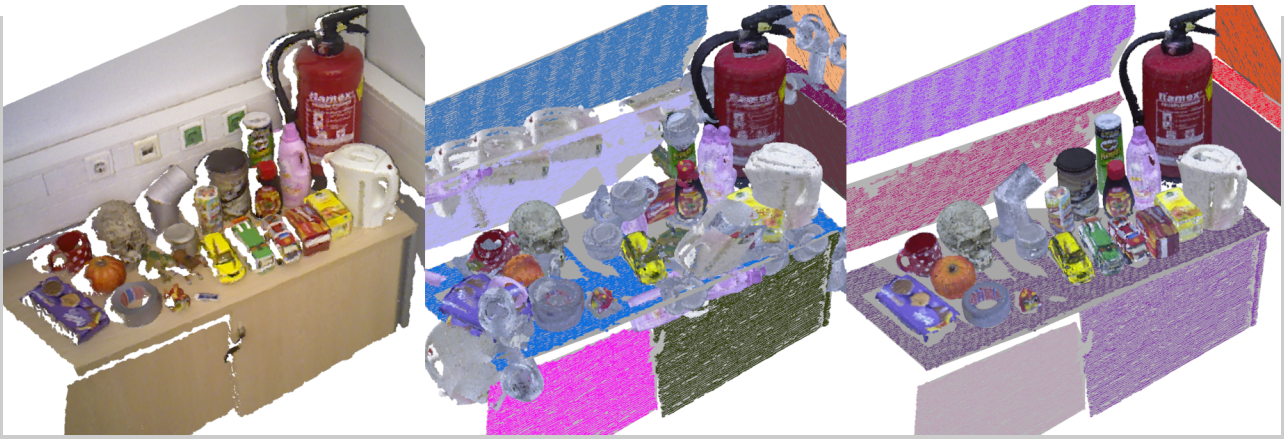

Fig. 12. Object instance recognition example with reconstructed 3D models: (Left) RGB-D point cloud (Middle) Object hypotheses (Right) Recognized objects displayed in the detected 6-DoF pose. With the exception of *horse*, *dino*, and *rubber*, all objects are correctly detected without any false positive.

[8] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon, "Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera." ACM Symposium on User Interface Software and Technology, October 2011.

[9] B. Curless and M. Levoy, "A volumetric method for building complex models from range images," in *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '96. New York, NY, USA: ACM, 1996, pp. 303–312.

[10] W. Kehl, N. Navab, and S. Ilic, "Coloured signed distance fields for full 3d object reconstruction," in *British Machine Vision Conference*, 2014.

[11] 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 Conference (RSS)*, 2013.

[12] M. Dimashova, I. Lysenkov, V. Rabaud, and V. Eruhimov, "Tabletop object scanning with an rgb-d sensor." 3rd Workshop on Semantic Perception, Mapping and Exploration (SPME), 2013.

[13] J. Sturm, E. Bylow, F. Kahl, and D. Cremers, "CopyMe3D: Scanning and printing persons in 3D," in *German Conference on Pattern Recognition (GCPR)*, Saarbrücken, Germany, September 2013.

[14] C. Tomasi and T. Kanade, "Detection and Tracking of Point Features," Carnegie Mellon University, Tech. Rep. CMU-CS-91-132, Apr. 1991.

[15] E. Rosten, R. Porter, and T. Drummond, "Faster and better: A machine learning approach to corner detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 32, pp. 105–119, 2010.

[16] J. Prankl, T. Mrwald, M. Zillich, and M. Vincze, "Probabilistic cue integration for real-time object pose tracking," in *Computer Vision Systems (ICVS)*. Springer Berlin Heidelberg, 2013.

[17] S. Fantoni, U. Castellani, and A. Fusiello, "Accurate and automatic alignment of range surfaces," in *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT)*, 2012, pp. 73–80.

[18] C. V. Nguyen, S. Izadi, and D. Lovell, "Modeling Kinect Sensor Noise for Improved 3D Reconstruction and Tracking." in *3DIMPVT*. IEEE, 2012, pp. 524–530.

[19] D. Huber and M. Hebert, "Fully Automatic Registration of Multiple 3D Data Sets," in *IEEE Computer Society Workshop on Computer Vision Beyond the Visible Spectrum(CVBVS 2001)*, December 2001.

[20] F. Tombari, S. Salti, and L. Di Stefano, "Unique signatures of Histograms for local surface description," in *Proc. 11th ECCV*, 2010.

[21] A. Aldoma, Z.-C. Marton, F. Tombari, W. Wohlkinger, C. Potthast, B. Zeisl, R. B. Rusu, S. Gedikli, and M. Vincze, "Tutorial: Point Cloud Library: Three-Dimensional Object Recognition and 6 DoF Pose Estimation," *Robotics & Automation Magazine, IEEE*, vol. 19, no. 3, pp. 80–91, 2012.

[22] A. Aldoma and M. Vincze, "Pose Alignment for 3D Models and Single View Stereo Point Clouds Based on Stable Planes," *3DIMPVT*, 2011.

[23] M. M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson surface reconstruction." in *Symposium on Geometry Processing*, ser. ACM International Conference Proceeding Series, A. Sheffer and K. Polthier, Eds., vol. 256. Eurographics Association, 2006, pp. 61–70.

[24] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm," in *ACM siggraph computer graphics*, vol. 21, no. 4. ACM, 1987, pp. 163–169.

[25] Z. Chen, J. Zhou, Y. Chen, and G. Wang, "3d texture mapping in multi-view reconstruction," in *Advances in Visual Computing*. Springer, 2012, pp. 359–371.

[26] A. Kasper, Z. Xue, and R. Dillmann, "The kit object models database: An object model database for object recognition, localization and manipulation in service robotics," *The International Journal of Robotics Research*, 2012.

[27] A. Aldoma, F. Tombari, J. Prankl, A. Richtsfeld, L. Di Stefano, and M. Vincze, "Multimodal cue integration through Hypotheses Verification for RGB-D object recognition and 6DoF pose estimation," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, 2013, pp. 2104–2111.

[28] T. Mörwald, J. Prankl, A. Richtsfeld, M. Zillich, and M. Vincze, "Blort - the blocks world robotic vision toolbox," in *Best Practice in 3D Perception and Modeling for Mobile Manipulation*, 2010.