In-hand Object Scanning via RGB-D Video Segmentation

Fan Wang[1] and Kris Hauser[1]

***Abstract*— This paper proposes a technique for 3D object scanning via in-hand manipulation, in which an object reoriented in front of a video camera with multiple grasps and regrasps. In-hand object tracking is a significant challenge under fast movement, rapid appearance changes, and occlusions. This paper proposes a novel video-segmentation-based object tracking algorithm that tracks arbitrary in-hand objects more effectively than existing techniques. It also describes a novel RGB-D in-hand object manipulation dataset consisting of several common household objects. Experiments show that the new method achieves 6% increase in accuracy compared to top performing video tracking algorithms and results in noticeably higher quality reconstructed models. Moreover, testing with a novice user on a set of 200 objects demonstrates relatively rapid construction of complete 3D object models.**

I. Introduction

The ability to build 3D models of the geometry and appearance of real-world objects has long been essential in many application scenarios such as art and design, manufacturing and packaging, augmented reality, and robotics. While early 3D model acquisition relied on multiple calibrated scanners or accurate motion apparatuses [1]–[3], the last decade has seen the introduction of low-cost RGB-D cameras like Microsoft Kinect and advances in feature detection and automatic feature matching algorithms that have made 3D object model acquisition easier than ever.

Among various 3D scanning techniques, in-hand scanning has attracted much research attention [4]–[7]. Methods that require an object to be stationary during the scan inherently carry the missing side problem [2], while in-hand scanning can reveal all sides of the object, making it possible to reconstruct the complete object without multiple model alignment in post-processing. The human hand is capable of intricate motion when manipulating objects through gripping, grasping and turning, so this is a natural way of presenting an object for scanning. It is also low-cost and flexible since it does not require expensive equipment and calibration. Many reconstruction pipelines from object-hand interaction require only an off-the-shelf RGB-D camera [4], [5].

Despite these attractive characteristics, 3D object modeling from in-hand object manipulation remains a challenging problem. To obtain a high-quality object model, the target object should be accurately separated from non-target objects. This separation is difficult since hands may be similar to the target object in terms of color, motion, or appearance, and furthermore, the target object itself undergoes significant changes in a short period of time.

*This work is partially supported by NSF CAREER grant #1253553.

F. Wang and K. Hauser are with the Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708, USA {fan.wang2, kris.hauser}@duke.edu

Fig. 1: Some successfully reconstructed objects from a 200-item test set, scanned by a novice user of our system.

This paper presents a 3D model acquisition pipeline from in-hand interaction. The pipeline includes a novel object tracking technique and a set of reconstruction and post-processing procedures. With this pipeline, a non-expert can scan arbitrary objects with only a hand-held single RGB-D camera and light manual annotation.

To evaluate our work, we constructed a public in-hand object manipulation dataset consisting of 13 objects from the publicly available YCB object set [8] being manipulated by hand in front of an RGB-D camera. Experiments on this dataset demonstrate that our method outperforms many state-of-the-art video segmentation algorithms in terms of tracking performance and results in higher quality 3D reconstructed models. We also asked a novice user to use our pipeline to scan 200 arbitrary items, requiring approximately 2.5 minutes of manual effort per object including scanning and annotation, and demonstrates the capability to produce high-quality reconstructed models, shown in Fig. 1.

II. Related Work

An object scanning pipeline commonly consists of object-background segmentation, multi-view registration, and registration refinements steps. Several in-hand scanning systems have been presented [1], [4], [7], [9], [10]. Weise et al. [9] demonstrated real-time system reconstruction of textured objects from in-hand manipulation using coarse to fine registration and online loop closure. Tzionas et al. [4] showed the possibility of reconstructing featureless objects from hand motion cues, by minimizing objective functions that consist of both visual correspondences and contact correspondences.

While the registration and refinement aspects of an object scanning system have been extensively studied [1], [2], [4],

[6], [9], little work has been done to address the unique challenges present in foreground - background segmentation in an in-hand manipulation task, despite its importance being commonly acknowledged [3], [6], [7], [11].

Existing in-hand scanning systems address the segmentation task with simple approaches. Hand-object segmentation are commonly addressed by skin pixel identification based on statistically color model [4], [9], or by having operators wear black gloves [1], [6], [10]. To remove other unwanted backgrounds, depth thresholding and black backgrounds are commonly used [1], [4]. A more sophisticated method is to perform segmentation in phase image as in [9] to remove background that remains stationary.

However, those methods are not robust enough to achieve highly accurate object-background segmentation. Color based skin-pixel segmentation, for example, can result in a high false negative rate with hands under different lighting conditions and a high false positive rate when objects being manipulated are close in color to human hands, as illustrated in Fig. 2. The phase image segmentation cannot remove non-stationary background objects such as hands and arms. The black glove method might be the most robust among all, but nonetheless requires only the hand and the object to be visible during the entire scanning, which can be difficult for the operator to accomplish.

Our pipeline uses similar registration approaches to prior work, but to address hand-object separation with more generality and to higher accuracy, we solve a semi-supervised binary video segmentation problem, tailored for the hand-object segmentation task. Our method can separate objects accurately from all unwanted backgrounds, including hands of various skin tones, as well as other stationary or non-stationary objects not of interest.

Recent research on video segmentation-based object tracking can be roughly categorized into unsupervised, semi-supervised, and human-in-the-loop approaches.

Unsupervised methods are fully automatic and require no human effort during run time [12]–[15]. While this approach is preferred when real-time tracking is necessary, a challenge is to propose a foreground hypothesis without specific knowledge on the actual object to track. Proposals are commonly formed from pre-learned models of "object-like" regions using static cues such as convex regions, pixel group similarities, or classification based on learning [16]–[18] as well as motion cues [15], [19], given that rigid foreground objects should move with coherent motion different from the background. Unsupervised segmentation for object scanning is difficult because pre-training is unlikely to apply to the large variety of arbitrary objects to be scanned, and motion cues from human hand movement can mislead the algorithm.

Semi-supervised and human-in-the-loop methods require varying amount of human interaction to specify the object of interest for higher accuracy [20]. In semi-supervised methods, the human provides an initial ground truth, typically for the first frame of a video sequence. This ground truth could be a segment mask [15], [21] or user scribbles annotating foreground and background [22], [23]. The algorithm then propagates this annotation segment to the unlabeled frames. Some semi-supervised methods use the first frame for fine-tuning a learned object model [21], [24], [25] and then perform per-frame segmentation. However, most semi-supervised methods use a propagation scheme to propagate the initial labels to successive frames [15], [19], [26].

Fig. 2: Skin regions identified by skin-pixel based segmentation described in [27]. This method cannot detect shaded areas of hand as well as accessories on hand (jewelry, nail polish, etc.). It also falsely includes object regions with colors close to human hands.

Fig. 3: The new sides problem poses a challenge for video segmentation algorithms that rely on coherent object appearance. Tracking a can of tomato soup with a hierarchical graph-based method [28] fails to grow the segmentation to newly appeared top of the can.

III. In-hand Object Reconstruction Pipeline

Our pipeline takes an RGB-D video as input, in which a user rotates the object in front of the camera. The user can perform multiple regrasps to reveal all sides of the object. Our method then uses a novel semi-supervised video segmentation technique, called BackFlow, to segment the object from the hands and background. These object segments are then registered to a global reference frame, and the overall model is post-processed to reduce noise.

A. BackFlow Video Segmentation Method

BackFlow is designed to handle many unique challenges for hand-object segmentation in RGB-D videos. For example, the **new sides problem** occurs because objects being manipulated quickly undergo significant shape and appearance changes. These changes are difficult to predict without prior knowledge of the object. Conventional video segmentation methods have difficulty maintaining tracking (Fig. 3). **Object-hand appearance similarity** is also quite common. Although the appearance of human hands varies from person to person, various skin-tone colors are commonly represented on the packaging. Additionally, the hand could also be wearing nail polishes or jewelry, which further complicates the identification of hand from the object. **Occlusion and large deformation** are especially prominent with human hands. Hands can occlude large portions of the object, and deform quickly and significantly. **Finger-object motion similarity** also tends to confuse algorithms that rely on distinct motion cues.

BackFlow derives new object appearance hypotheses by deduction, rather than through foreground coherence. It preferentially tracks background pixels and derives foreground accordingly through a graph-based framework. Essentially, it assumes that what is not background is likely foreground, and it attempts to track and detect the disappearance and emergence of the background. The rationale behind this approach is that hands and background objects exhibit stronger spatiotemporal coherence compared with the target object, and therefore are the better candidate to be tracked.

BackFlow requires no prior training and forms initial object proposals based on color GMMs learned in the first segment. The proposals are adjusted during run-time with more available frames. This method is most effective when the appearance and shape of the foreground object evolve quickly while the background remains relatively stable.

The four main steps of BackFlow are described below:

1) Prepossessing and initialization: BackFlow first eliminates unnecessary background tracking. A depth cut-off on the color images is performed that eliminates areas more than 1 m away from the camera. Then it requires a human to initialize *sure background* pixels, either by providing background scribbles or a segment mask. These annotations serve as hard constraints in graph-cutting for the first frame.

2) Background pixel propagation: Next, we use optical flow [29] to propagate raw background pixels between two successive frames. The flow is cross-examined by performing backward flow, and background pixels with the inconsistent flow are dropped. Since most optical flow inconsistencies occur near the motion edge, this helps prevent background labels from drifting into the foreground.

Superpixels are employed to guide the background propagation and to increase robustness against drift. Specifically, we perform SLIC segmentation [30] with N = 20,000 superpixels on frames with 640*480 resolution. When propagating background pixels from frame t to frame t+1, all frame t superpixels that contain raw background pixels are marked as the background superpixels. We track the flow of all pixels within the background superpixels to the second frame and label frame t+1 superpixels that receive enough propagated background pixels as new background superpixels. The use of superpixels provides additional color information and prevents drifts into neighboring groupings that differ in color. This measure also replenishes background labels within a superpixel under the assumption that all pixels within a superpixel likely belong to the same label.

3) Segmentation with relaxed foreground derivation: Step 3 performs graph-based segmentation after the background pixels have been propagated to the current frame. Pixels form vertices connected to neighbors and both background and foreground labels by weighted edges. The segmentation is formulated as an energy minimization problem in which the energy function has the form:

$$E(L) = \sum_{p \in \mathcal{P}} D_p(L_p) + \gamma \sum_{(p,q) \in \mathcal{N}} V_{(p,q)}(L_p, L_q), \quad (1)$$

where $L = \{L_p | p \in \mathcal{P}\}$ is a labeling of image $\mathcal{P}$, $D_p(\cdot)$ is the data penalty function assigning labels to the vertex, $V_{(p,q)}$ is the smoothness term, and $\mathcal{N}$ is a set of all pairs of neighboring pixels [23]. γ is a parameter chosen to balance data and smoothness terms. The minimum-energy cut of the graph is therefore a partition of the vertices into two disjoint sets, solved by min-cut/max-flow algorithms [23].

We use Gaussian Mixture Models (GMMs) to describe background and foreground distributions similar to [22]. We modify standard graph segmentation algorithms to encourage foreground propagation and to rely more heavily on motion cues in the presence of drastic appearance changes.

Background GMMs with K=8 components are learned on the first frame and used throughout the video sequence, and foreground GMMs are learned on a sample of raw pixels in the previous foreground segments. Specifically, we store past foreground pixels, and after each new frame is segmented, we downsample the foreground set by 90% and add 10% new foreground pixels; this is to ensure a stable background proposal and a smoothly transitioning foreground proposal as the object evolves.

In BackFlow, we formulate initial data penalty terms similar to Grabcut [22], which indicate the log likelihood of a pixel belonging to the foreground and background GMMs, respectively:

$$\begin{aligned}\tilde{D}(\alpha_p, k_p, \underline{\theta}, z_p) = &- \log \pi(\alpha_p, k_p) + \frac{1}{2} \log \det \sum (\alpha_p, k_p) \\ &+ \frac{1}{2}[z_p - \mu(\alpha_p, k_p)]^{\intercal} \sum (\alpha_p, k_p)^{-1} [z_p - \mu(\alpha_p, k_p)],\end{aligned} \quad (2)$$

where α_p indicates background or foreground assignments and has value of 0 or 1 in hard segmentation. $\underline{\theta}$ stands for the GMM parameters. k_p is the GMM component assigned to the pixel and $k_p \in \{1, ..., K\}$ where K is the number of components in the GMMs. z_p the a pixel color in RGB colour space. The weights π, mean μ and covariance Σ are the parameters for the Gaussians from the foreground and background distributions.

The data cost for each node is by default $D \leftarrow \tilde{D}$, except for the following exceptions. For nodes already associated with background labels, we assign D to a high constant penalty to be connected to the foreground and zero penalty to be connected to the background. For unlabeled nodes, we further add a comparison step in which we equalize data penalties assigned to both labels if a pixel only has a slightly lower probability belonging to the foreground than the background proposal. The background-foreground assignment of such node is therefore determined by the assignments of neighborhood pixels that have a high interacting potential to the node. Specifically, if

$$\tilde{D}(\alpha_{p=1}, k_p, \underline{\theta}, z_p) > \tilde{D}(\alpha_{p=0}, k_p, \underline{\theta}, z_p), \quad (3)$$

and

$$\tilde{D}(\alpha_{p=1}, k_p, \underline{\theta}, z_p) - \tilde{D}(\alpha_{p=0}, k_p, \underline{\theta}, z_p) < N, \quad (4)$$

then

$$D(\alpha_{p=1}, k_p, \underline{\theta}, z_p) \leftarrow \tilde{D}(\alpha_{p=0}, k_p, \underline{\theta}.z_p), \quad (5)$$

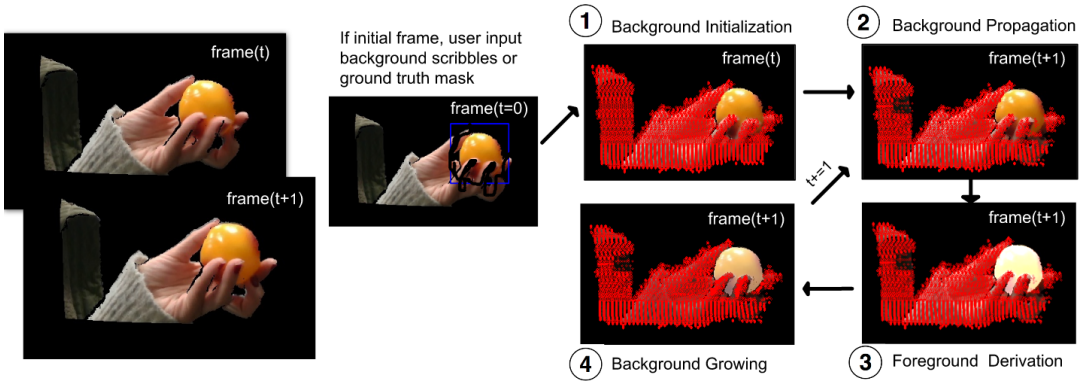

Fig. 4: BackFlow takes user annotation of background or ground truth segment on the first frame and propagates labels as follows. 1) Initialize background labels (superpixels) from user input. 2) Propagate background labels from frame t to frame t+1 through optical flow. 3) Perform graph-based foreground derivation. 4) Remove background labels in segmented foreground and grow background labels to neighbourhood superpixels.

where N is the relaxation constant. Experiments find that the segmentation accuracy is not greatly impacted by the value of N for $8 < N < 16$.

The introduction of the relaxation constant encourages the inclusion of the newly appeared foreground that is adjacent to the existing foreground. While this measure would otherwise introduce false positives if the background is not densely labeled, the majority of the background area is already correctly labeled. We also observed that the difference in data penalties for newly appeared background (e.g., hands, arms) are usually much greater than the relaxation constant.

A commonly used smoothness term (e.g., in Grabcut) is:

$$V(\underline{\alpha}, z) = \sum_{(m,n)\in\mathbf{C}} I[\alpha_n \neq \alpha_m] \exp(-\beta||Z_m - Z_n||^2)$$

z is the given image data, $\underline{\alpha}$ is the background or foreground assignments for each pixel, $\mathbf{C}$ is the set of all neighbouring pixels, $||Z_m - Z_n||^2$ is the Euclidean distance between neighbouring pixels in RGB color space. β measures the inverse expectation of contrast within the image.

In BackFlow we introduce motion cues into the smoothness term. To encourage consistent labeling for areas with coherent motions, we use optical flow along with color dissimilarities for smoothness term calculations. We convert the flow vectors into images in HSV color space where the Hue and Value are computed from the angle and magnitude of the flow. The smoothness penalties are then taken as the smaller of the discontinuity in color and flow:

$$V(\underline{\alpha}, z) = \sum_{(m,n)\in\mathbf{C}} I[\alpha_n \neq \alpha_m] \min(\exp(-\beta_z||Z_m - z_n||^2), \exp(-\beta_f||F_m - F_n||^2)).$$

The expectation value β_z and β_f are calculated respectively for RGB image and the flow image. In our implementation we use $\gamma = 50$ and $N = 10$. The cut is then performed using Min-Cut/Max-Flow algorithms. The largest connected component in the result becomes the foreground segment.

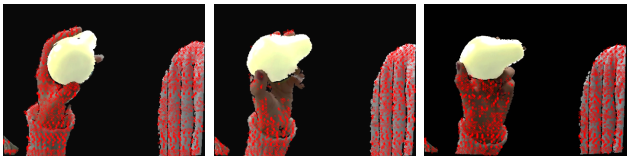

Fig. 5: Background labels (marked as red dots) grow gradually into the previously unseen palm while maintaining separation from the foreground object.

4) Removing and growing background labels: Finally, we extend background tracking into the newly appeared background. After segmentation of each frame, all superpixels that are spatially adjacent to the existing background and are not segmented as foreground become background superpixels and will be tracked in the next frame as shown in Fig. 5.

B. 3D Reconstruction

The segmented frames are registered to produce the final model. To avoid redundant information and bad frames with severe motion blur/noise, keyframes are selected for registration. Pairwise registration between keyframes is performed using sparse feature-point matching followed by dense Iterative Closest Point (ICP) [31] alignment. Finally, we run pose-graph alignment and postprocessing.

By default, we select every 10th frame as the keyframe if it either 1) matches enough feature points from the previous keyframe or 2) its point cloud can be matched above a certain percentage with the previous keyframe with ICP. Otherwise, we try to match the previous keyframe with the 9th frame, 8th frame ... until a match is found. If no match exists, we will select the 10th frame and continue with the process.

For pairwise registration we extract SIFT (Scale-Invariant Feature Transform) [32] features from consecutive keyframes and match feature pairs through a 2D RANSAC (Random sample consensus) homography process, and use the depth correspondence of the matched feature point to find the best 3D rigid body transformation between the two frames. If insufficient matching features are found, dense ICP is performed between two point clouds to estimate the transforma-

Fig. 6: RGB-D in-hand object manipulation dataset contains RGB-D video of 13 non-duplicate items from YCB food category, including Pringles, mustard, Cheez-It, sugar, Spam, tomato soup, banana, pear, plum, strawberry, orange, lemon and Jell-O.

TABLE I: Segmentation Results

	OSVOS	SFL	HVS	SPBS	BackFlow
IoU	87.97%	29.61%	52.05%	31.52%	**93.52%**
FP	4.17%	50.59%	**1.7%**	49.99%	2.12%
FN	7.85%	13.90%	46.29%	18.48%	**4.36%**

TABLE II: Hausdorff Distance measurements w.r.t. bounding box size of the reconstructed models

	Max distance	Mean distance	RMS
BackFlow	0.195	0.014	0.017
OSVOS	0.427	0.024	0.031

tion. We use the colored point cloud registration by Park et al. [33] implemented in Open3D [34]. This ICP variant jointly optimizes both point-to-plane distances and minimizes the error of projected color. Pose graph optimization is used to obtain the final registration transforms in the global frame, with methods described in [35].

For postprocessing, the aligned colored point clouds are stored in voxel grids of dimension 2mm, and each voxel is optimized to reject color outliers, and grids with less than 2 points are not used. Registered point clouds then go through Poisson surface reconstruction [36] to create watertight surfaces, and isolated components whose diameter is smaller than 10% of the mesh diameter are removed to achieve the final models.

IV. Experiments

A. RGB-D in-hand object manipulation dataset

Although there are several popular video segmentation datasets, there is no existing in-hand object scanning RGB-D video dataset with dense pixel level annotations. To systematically evaluate our methods and to aid the research community, we introduce the RGB-D In-hand Object Manipulation Dataset (Fig. 6). This dataset contains 13 sequences of in-hand manipulation of objects from the YCB object set [8], recorded with an Intel RealSense SR300 RGB-D camera. Each sequence ranges from 300 to 800 frames in length (filmed at 30 fps) and contains in-hand manipulation of the objects revealing all sides. The dataset is complete with color images, depth cut-off images, and depth mages. The pixel-wise annotation aligned to the depth cut off image is provided every 10 frames. The dataset and results can be downloaded at the RGB-D In-hand Object Manipulation website `https://www.rgbdinhandmanipulation.com`.

The performance of BackFlow is compared to other state-of-the-art segmentation methods in Tab. I. OSVOS is a semi-supervised fully-convolutional neural network (FCNN) that was one of the top-performers on DAVIS 2017 [21]. SFL indicates the unsupervised FCNN technique SegFlow [15]. Both OSVOS and SFL were pre-trained on the RGB-D Object Dataset [37] while OSVOS was additionally fine-tuned on the initial segment of each sequence. HVS is the hierarchical graph-based video segmentation of [28]. SPBS indicates skin-pixel based segmentation, which is a GMM method described in [27], trained on the UCI skin segmentation dataset [38].

These experiments separate each object handling sequence into sub-sequences of 100 images. A ground truth annotation is given on the first frame of each sub-sequence, and the segmentation accuracy is tested on the remaining 99 frames. We measure the intersection over union (IoU), areas of false positives over union (FP) and areas of false negatives over union (FN). BackFlow outperforms other state-of-the-art approaches by almost 6% percent in overall accuracy and achieves lower false positive and false negative rate. Per-sequence results are shown in Fig. 7.

The reconstructed models of several objects from the dataset are shown in Figure 8. The textured (color or geometry) objects are reconstructed complete with all sides, except when the surface material (e.g., black, highly reflective) cannot be captured by a depth camera. Our reconstruction pipeline currently cannot obtain a complete model with symmetrical and texture-less objects, such as pear and orange, and the registered point clouds appear as incomplete "shell".

Fig. 9 shows some example of unprocessed point cloud reconstruction from BackFlow against the next-best performer OSVOS. To qualitatively measure the quality of the reconstructed models, we further compare all 13 registered point clouds constructed from BackFlow and OSVOS to the ground truth Poisson meshes provided in the YCB dataset. The evaluation metric used is the one-sided Hausdorff Distance implemented in MeshLab. To measure it, the registered point cloud and the YCB mesh are aligned using manual key points matching (we use the same rigid transformation for both registered point clouds), then the point cloud is sampled to 20,000 points and for each sample, the Hausdorff Distance to the closest point over the YCB mesh is computed.

Shown in Tab. II, the Hausdorff distance measurements for BackFlow reconstructed models are around 50% smaller than OSVOS in all categories including max distance, mean distance, and RMS. The max distance in BackFLow is caused by failure to segment out a human finger in parts of the banana sequence, the artifact is reflected on the reconstruct (see Fig. 8), yet the max distance is still much smaller than

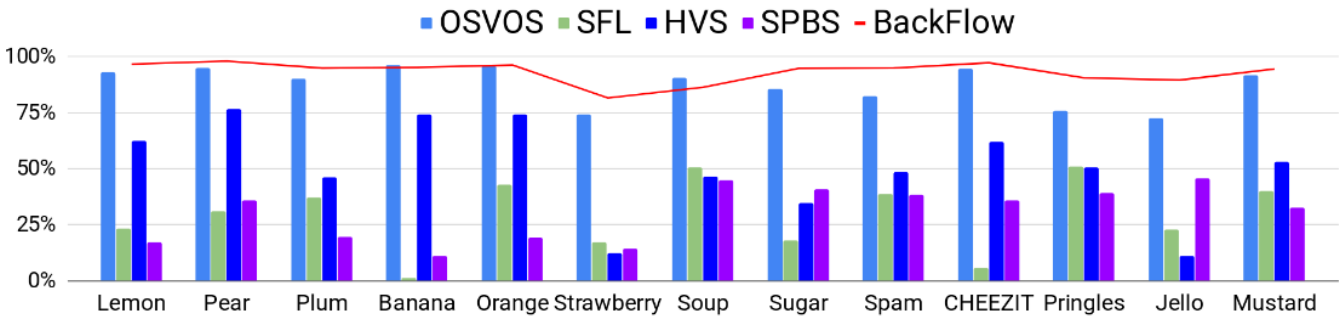

Fig. 7: Per sequence accuracy (IoU) comparison of BackFlow to other state-of-the-art video segmentation methods.

Fig. 8: Reconstructed models from BackFlow segmentation results.

Fig. 9: Comparison of reconstructions from OSVOS and BackFlow.

that of OSVOS. While BackFlow only improves segmentation performance by 6%, the resulting reconstructions are of significantly better quality, as these segments maintain a better consistency and have a lower false positive rate.

B. Novice scanning of many items

To evaluate the ease of use and robustness of the pipeline when handled by a novice user, we presented our software to a volunteer who is unfamiliar with 3D object modeling. 200 grocery items were purchased satisfying the following criteria: 1) the size of the object should be easily manipulated with a single hand, 2) the object does not contain large black or transparent areas, which cannot be captured with a structured light RGB-D camera, 3) the object is textured or non-symmetric, 4) objects should not significantly deform during manipulation.

The volunteer was taught how to use the software and was told to reveal all sides of the object. No other instruction on how to manipulate the objects was given. We record the total amount of time needed, including the instruction time, and several small breaks, for the volunteer to finish scanning the objects. The volunteer then annotates every 100 frames of the scan sequence as initial labels.

On average, the volunteer took 102 seconds to scan each item, plus another 36 seconds for labeling (averaging 4.9 frames per item). Segmentation and reconstructions then automatically run to generate object reconstructions. The total labor time used for reconstructing 200 models is 7 hours and 40 minutes, or 1 fps for segmentation and reconstruction on a CPU computer.

As shown in Fig. 1, the pipeline is robust enough to generate many visually high-quality reconstructions with no hands or any other background objects found in any reconstructed model. There are also failed reconstructions, which are most frequently caused by misaligned frames during registration. Better registration techniques are likely to improve the results further.

V. Conclusion

This paper presented an in-hand object scanning pipeline and benchmarking dataset. Experiments demonstrate that the novel BackFlow video segmentation technique leads to improved tracking accuracy for fast changing objects when compared to other state-of-the-art methods, which leads to better reconstruction of 3D models. The only equipment needed by the method is an RGB-D camera and standard PC. Moreover, the software is easy to use for novices, taking approximately 2 minutes of manual input per object. Future work may improve registration of symmetric or featureless objects by exploring motion cues from finger movements. We would also like to better estimate appearance information of our reconstructed models, which currently exhibit artifacts from changing lighting conditions between frames.

REFERENCES

[1] S. Rusinkiewicz, O. Hall-Holt, and M. Levoy, "Real-time 3d model acquisition," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 438–446, July 2002.

[2] F. Bernardini and H. Rushmeier, "The 3d model acquisition pipeline," *Computer Graphics Forum*, vol. 21, no. 2, pp. 149–172, 2002.

[3] M. Krainin, P. Henry, X. Ren, and D. Fox, "Manipulator and object tracking for in-hand 3d object modeling," *Int. J. Rob. Res.*, vol. 30, no. 11, pp. 1311–1327, 2011.

[4] D. Tzionas and J. Gall, "3d object reconstruction from hand-object interactions," *CoRR*, vol. abs/1704.00529, 2017.

[5] J. Prankl, A. Aldoma, A. Svejda, and M. Vincze, "Rgb-d object modelling for object recognition and tracking," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 96–103.

[6] T. Weise, T. Wismer, B. Leibe, and L. V. Gool, "In-hand scanning with online loop closure," in *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, 2009, pp. 1630–1637.

[7] P. Panteleris, N. Kyriazis, and A. A. Argyros, "3d tracking of human hands in interaction with unknown objects," in *BMVC*, 2015.

[8] B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, "Benchmarking in manipulation research: Using the yale-cmu-berkeley object and model set," *IEEE Robotics Automation Magazine*, vol. 22, no. 3, pp. 36–52, 2015.

[9] T. Weise, B. Leibe, and L. V. Gool, "Accurate and robust registration for in-hand modeling," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.

[10] 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.

[11] S. Sridhar, F. Mueller, M. Zollhoefer, D. Casas, A. Oulasvirta, and C. Theobalt, "Real-time joint tracking of a hand manipulating an object from rgb-d input," in *Proceedings of European Conference on Computer Vision (ECCV)*, 2016.

[12] Y. G. Lee, Z. Tang, J. N. Hwang, and Z. Fang, "Inter-camera tracking based on fully unsupervised online learning," in *2017 IEEE International Conference on Image Processing (ICIP)*, 2017, pp. 2607–2611.

[13] F. Xiao and Y. J. Lee, "Track and segment: An iterative unsupervised approach for video object proposals," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 933–942.

[14] W. Wang, J. Shen, and F. Porikli, "Saliency-aware geodesic video object segmentation," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3395–3402.

[15] J. Cheng, Y. H. Tsai, S. Wang, and M. H. Yang, "Segflow: Joint learning for video object segmentation and optical flow," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 686–695.

[16] G. Huang, C.-M. Pun, and C. Lin, "Unsupervised video co-segmentation based on superpixel co-saliency and region merging," *Multimedia Tools Appl.*, vol. 76, no. 10, pp. 12 941–12 964, May 2017.

[17] H. Ramadan and H. Tairi, "Automatic human segmentation in video using convex active contours," in *2016 13th International Conference on Computer Graphics, Imaging and Visualization (CGiV)*, 2016, pp. 184–189.

[18] M. Khodabandeh, S. Muralidharan, A. Vahdat, N. Mehrasa, E. M. Pereira, S. Satoh, and G. Mori, "Unsupervised learning of supervoxel embeddings for video segmentation," in *2016 23rd International Conference on Pattern Recognition (ICPR)*, 2016, pp. 2392–2397.

[19] Y. H. Tsai, M. H. Yang, and M. J. Black, "Video segmentation via object flow," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 3899–3908.

[20] DAVIS2017, "Benchmark video object segmentation on davis," http://davischallenge.org/soa_compare.html, accessed: 2018-09-12.

[21] S. Caelles, K. K. Maninis, J. Pont-Tuset, L. Leal-Taix, D. Cremers, and L. V. Gool, "One-shot video object segmentation," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5320–5329.

[22] C. Rother, V. Kolmogorov, and A. Blake, ""grabcut": Interactive foreground extraction using iterated graph cuts," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 309–314, Aug. 2004.

[23] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision," pp. 1124–1137, 2004.

[24] F. Perazzi, A. Khoreva, R. Benenson, B. Schiele, and A. Sorkine-Hornung, "Learning video object segmentation from static images," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 3491–3500.

[25] P. Voigtlaender and B. Leibe, "Online adaptation of convolutional neural networks for video object segmentation," in *BMVC*, 2017.

[26] S. Poullot and S. Satoh, "Vabcut: A video extension of grabcut for unsupervised video foreground object segmentation," in *2014 International Conference on Computer Vision Theory and Applications (VISAPP)*, vol. 2, 2014, pp. 362–371.

[27] V. Vezhnevets, V. Sazonov, and A. Andreeva, "A survey on pixel-based skin color detection techniques," in *IN PROC. GRAPHICON-2003*, 2003, pp. 85–92.

[28] M. Grundmann, V. Kwatra, M. Han, and I. Essa, "Efficient hierarchical graph-based video segmentation," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010, pp. 2141–2148.

[29] G. Farnebäck, "Two-frame motion estimation based on polynomial expansion," in *Image Analysis*, J. Bigun and T. Gustavsson, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 363–370.

[30] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Ssstrunk, "Slic superpixels compared to state-of-the-art superpixel methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274–2282, 2012.

[31] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[32] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[33] J. Park, Q. Zhou, and V. Koltun, "Colored point cloud registration revisited," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017, pp. 143–152.

[34] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," *arXiv:1801.09847*, 2018.

[35] R. Kmmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G2o: A general framework for graph optimization," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 3607–3613.

[36] M. Kazhdan and H. Hoppe, "Screened poisson surface reconstruction," *ACM Trans. Graph.*, vol. 32, no. 3, pp. 29:1–29:13, July 2013.

[37] K. Lai, L. Bo, X. Ren, and D. Fox, "A large-scale hierarchical multi-view rgb-d object dataset," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 1817–1824.

[38] R. Bhatt, "Skin segmentation data set," https://archive.ics.uci.edu/ml/datasets/skin+segmentation, accessed: 2018-09-12.