A ROBUST AUTOMATIC OBJECT SEGMENTATION METHOD FOR 3D PRINTING

Tzu-Kuei Huang Ying-Hsuang Wang Ta-Kai Lin Yung-Yu Chuang

National Taiwan University

ABSTRACT

3D printing has become an important and prevalent tool. Image-based modeling is a popular way to acquire 3D models for further editing and printing. However, exiting tools are often not robust enough for users to obtain the 3D models they want. The constructed models are often incomplete, disjoint and noisy. This paper proposes a robust automatic method for segmenting an object out of the background using a set of multi-view images. With the segmentation, 3D reconstruction methods can be applied more robustly. The segmentation is performed by minimizing an energy function which incorporates color statistics, spatial coherency, appearance proximity, epipolar constraints and back projection consistency of 3D feature points. It can be efficiently optimized using the min-cut algorithm. Experiments show that the proposed method can generate better models than some popular systems.

Index Terms— 3D printing, object segmentation.

1. INTRODUCTION

The cost of 3D printing has decreased significantly recently. It greatly shortens the gap between the prototyping and manufacturing stages and also enables customized/personalized manufacturing. Ordinary users can now fabricate their own models at home using low-cost 3D printers. Accompanied by the popularity of 3D printing, there is a growing demand for computational tools that enable users with little or no manufacturing experience to design and print objects in 3D.

Image-based modeling, generating a 3D model from a set of images, is a popular way for users to acquire the 3D models of real-world objects. The acquired model can be further edited and 3D-printed. Conventional methods for geometry acquisition from images often consists of three steps: structure from motion, multi-view stereo and surface reconstruction [1]. Structure from motion estimates camera parameters for images and generates a set of sparse 3D points [2]. Multi-view stereo generates a denser point cloud by finding visual correspondences and triangulation [3]. Finally, surface reconstruction takes the dense point cloud and produces a globally consistent 3D model [4, 5].

This work was partly supported by grants MOST104-2218-E-002-015 and MOST104-2628-E-002-003-MY3.

Responding to the strong demand, there are quite a few free and commercial programs for 3D model acquistion from images, such as AutoDesk 123D Catch [6] and Multi-View Environment [1]. Unfortunately, even with these systems, acquiring 3D models from images is still not an easy process for ordinary users. The models obtained using existing systems could still suffer from the problems with incomplete, disjoint, noisy models and unwanted background (see Figure 1(f) for examples). The recently proposed MobileFusion system addresses this problem to some degree [7]. However, its goal is to fulfill the whole process on the mobile devices. Thus, the generated 3D models are often of low quality. Our goal is to synthesize 3D models with sufficient quality from a set of images and the reconstruction process can be executed offline on other platforms.

This paper proposes a robust automatic method for segmenting an object out of the background using a set of multi-view images. With the segmention, 3D reconstruction methods can be applied more robustly. Image segmentation is inherently a problem requiring user inputs because, without user inputs, it is impossible to identify the foreground object the user intends to extract. Thus, most segmentation methods are interactive and require some forms of user hints, such as scribbles [8] or bounding boxes [9]. Our method achieves automatic segmentation because, in our application, the images themselves likely reveal user's intention. For the application of acquiring geometry from images, users often take photographs around the object; and in each image, the object often locates at the center and the image often contains the whole object. Our method utilizes these properties for automatic segmentation and formulates a graphical model incorporating color statistics, spatial coherency, appearance proximity, epipolar constraints and back projection consistency of 3D points. With the segmentation, we generate 3D model using the simple visual hull method. More advanced 3D reconstruction methods could be applied for obtaining better models from the segmentation, but it is not the focus of the paper.

2. AUTOMATIC OBJECT SEGMENTATION

The input is a set of N images, $\{I_1, \ldots, I_N\}$, capturing the same object from different views. Figure 1(a) shows an example of four images from a set of 24 images. The goal is to separate the object from the background in each image. That is,

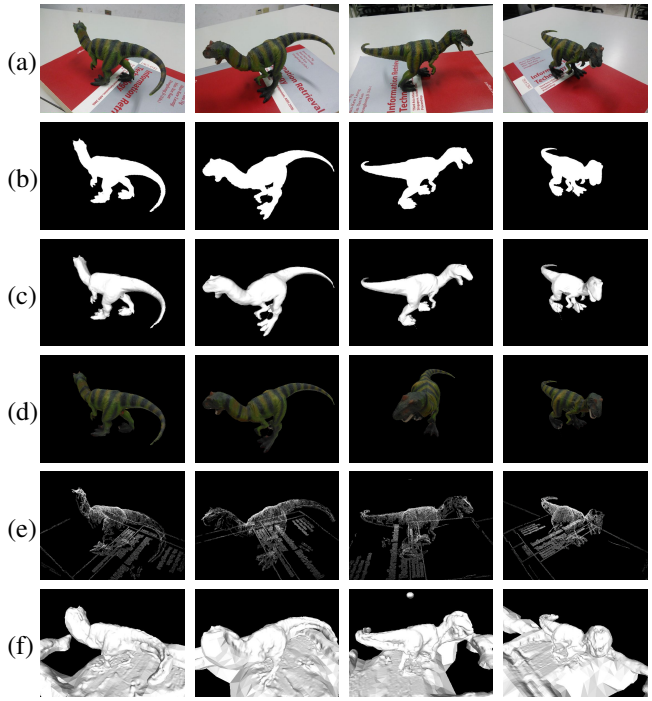

Fig. 1: Given a set of 24 images (four of them shown in (a)), our method automatically isolates the object from the background (b). With the segmentation results, a 3D model (c) is reconstructed using the visual hull method. The model can be texture mapped using the input images, as shown in (d). A state-of-the-art method uses the multi-view stereo method PMVS [3] to extract a point cloud (e) from this set of images, followed by Poisson surface reconstruction [4] for obtaining the 3D model (f). It has several problems.

for each pixel p, we label it as either the foreground (F) or the background (B). We utilize color statistics, spatial coherency and 3D constraints for solving this binary labeling problem. Note that we still assume that there is sufficient distinction between the foreground and background in their appearance. Figure 2 illustrates the flow of the proposed method.

2.1. Preprocessing

To reduce the problem size, we first over-segment the images into superpixels using the SLIC algorithm [10]. We denote the set of superpixels for the image I_k as S_k. The set of all superpixels is denoted as $\mathbf{S} = \cup_{k=1}^{N} S_k$. For each superpixel $s_i \in \mathbf{S}$, its color c_i is calculated as the average of all pixels' colors in s_i and its position x_i is the average position of all pixels in s_i. The use of superpixels not only speeds up the computation, but also allows more effective labeling because they ensure that pixels with similar colors and positions are labeled in the same way. The segmentation problem is then reduced to find the best label assignment Φ which assigns the label F or B to each superpixel $s_i \in \mathbf{S}$, i.e., $\Phi(s_i) \in \{F, B\}$.

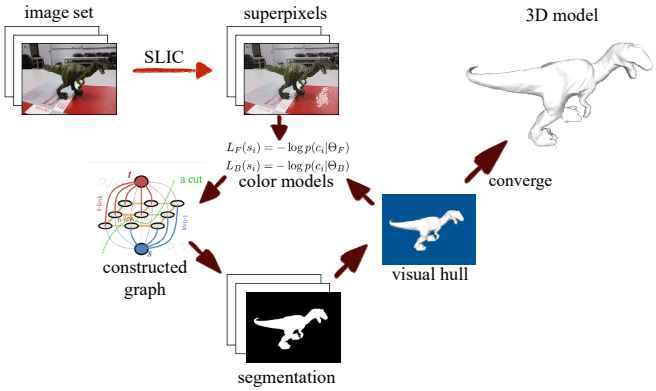

Fig. 2: The flow of the proposed method.

In preprocessing, we also perform structure from motion using VisualSFM [2] to obtain the camera parameters for each image and a set of sparse 3D points $\mathbf{Q}$, corresponding to the salient feature points in images. The extracted 3D information not only provides reliable correspondences for improving labeling, but also facilitates reconstruction of a visual hull for effective culling of the background.

2.2. Energy terms

There are totally five terms in the energy function, each corresponding to a property of labeling we would like to impose.

Color likelihood $\mathbf{E}_c$. We use Gaussian mixture models (GMM) as color models for the foreground and the background. A Gaussian mixture model is a probability function with K Gaussians:

$$p(x|\Theta) = \sum_{k=1}^{K} w_k G(x|\mu_k, \Sigma_k), \tag{1}$$

where G is the Gaussian function and Θ is the set of parameters consisting of weights w_k, means μ_k and covariance matrices Σ_k. By collecting a set of foreground and background samples, the parameters of GMMs can be estimated using the EM algorithm.

If we already have the segmentation results, we can simply use the color samples in the foreground region to estimate the parameters Θ_F for the foreground GMM model, and the background region for background parameters, Θ_B. For initializing the foreground and background segmentation, we utilize the assumption that, in all images, the object is fully captured and placed at the centers of images. Since we have the camera parameters for each image, we can obtain its viewing frustum. We obtain the visual hull by finding the intersection of viewing frustums of all images. By projecting the visual hull back to each view, we have a rough initial segmentation for the foreground and the background and use them for obtaining initial color models. The models will be refined with better segmentation results and the segmentation results

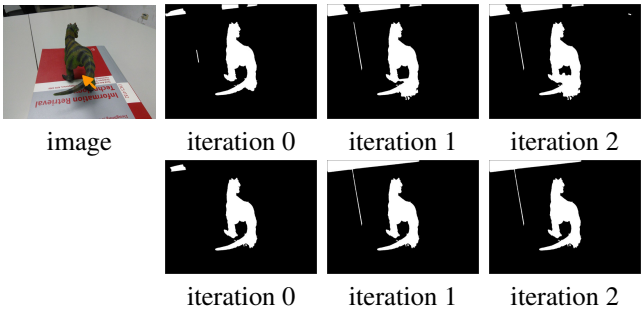

Fig. 3: The top row shows results without the appearance coherence term while the bottom row shows the ones with the term. With the appearance coherence term, the region between dinosaur's legs is labeled more accurately.

will benefit from better color models. We then denote the likelihood as:

$$L_F(s_i) = -\log p(c_i|\Theta_F), \tag{2}$$
$$L_B(s_i) = -\log p(c_i|\Theta_B), \tag{3}$$

where Θ_F and Θ_B are the parameters for the foreground and background GMM models. The color likelihood term can then be intuitively defined as

$$\mathbf{E}_c(s_i) = \begin{cases} \frac{L_F(s_i)}{L_F(s_i)+L_B(s_i)} & \text{if } \Phi(s_i) = F \\ \frac{L_B(s_i)}{L_F(s_i)+L_B(s_i)} & \text{if } \Phi(s_i) = B, \end{cases} \tag{4}$$

Spatial coherence $\mathbf{E}_s$. Neighboring superpixels tend to have similar labels due to spatial coherence within images. The label incoherence could however be tolerated if the colors of neighboring superpixels are not similar. We measure the similarity of two superpixels by their color discrepancy,

$$\phi(s_i, s_j) = \exp(\gamma \|c_i - c_j\|_2^2). \tag{5}$$

We denote the spatial neighbors of a superpixel s_i as $\mathbf{N}^s_{s_i}$. The spatial coherence term can then be defined as

$$\mathbf{E}_s(s_i, s_j) = \begin{cases} \phi(s_i, s_j) & \text{if } s_j \in \mathbf{N}^s_{s_i}, \Phi(s_i) \neq \Phi(s_j) \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

With this term, neighboring superpixels with similar appearance will be strongly bound and tend to be assigned with the same label. On the other hand, neighboring superpixels could still be assigned with different labels if there is a significant appearance discrepancy between them.

Appearance coherence $\mathbf{E}_a$. The above term only encourages coherent labeling between adjacent superpixels. This term considers nearby but not necessarily adjacent superpixels and further encourages label coherence between nearby superpixels with similar appearance. We denote the appearance neighbors of a superpixel s_i as $\mathbf{N}^a_{s_i}$, which contains the top K_s most similar superpixels within the region whose center is x_i with a user-specified radius r. The appearance coherence term is then defined as

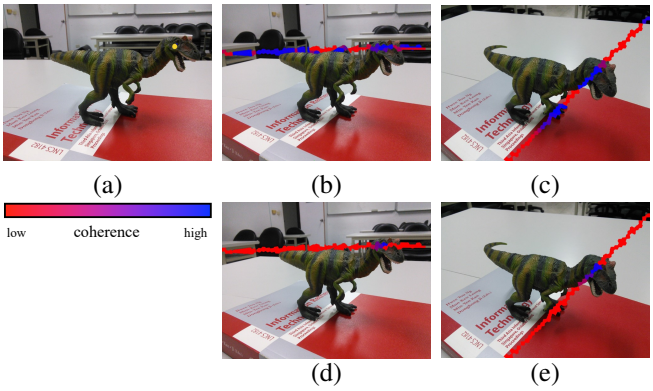

Fig. 4: The epipolar color coherence term along the epipolar lines corresponding to dinosaur's eye (the yellow point in (a)). With only color similarity, there are many false matches in (b) and (c). With rough depth estimation, the label coherence can be imposed correctly in (d) and (e).

$$\mathbf{E}_a(s_i, s_j) = \begin{cases} \phi(s_i, s_j) & \text{if } s_j \in \mathbf{N}^a_{s_i}, \Phi(s_i) \neq \Phi(s_j) \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

With this term, label assignment can be propagated among similar superpixels more effectively. Figure 3 demonstrates the effects of this term. In this example, the region between dinosaur's legs contains soft shadow whose darker part is similar to parts of the dinosaur in color. Without the appearance coherence term $\mathbf{E}_a$, along iterations, because of the spatial coherence term, the lighter part of the shadow also gradually becomes parts of the dinosaur. With $\mathbf{E}_a$, the lighter shadowed background could find more similar colors from further background regions. Thus, it is correctly labeled as the background and influences the darker shadow.

Epipolar color coherence $\mathbf{E}_e$. According to the theory of multiple view geometry [11], for each superpixel s_i, its corresponding superpixel on the other image I_k should locate on its epipolar line $l^k_{s_i}$. One option would be to associate s_i with each superpixel s_j on $l^k_{s_i}$ with their similarity as the cost, i.e., $\mathbf{E}_e(s_i, s_j) = \phi(s_i, s_j)$. This way, however, many false matches could hinder the correct segmentation. Figure 4(b) and (c) show the label coherence requirements of superpixels along epipolar lines in two neighboring views for the superpixel containing the dinosaur's eye (the yellow point in Figure 4(a)). There are many false matches that are assigned with high coherence requirement (blue color). For example, the chairs in the background and the stripes of the dinosaur have the similar color to the eye. Their labels are incorrectly tied with the eye's label this way. Similar to Campbell et al. [12], we use the depth estimation to resolve the ambiguity. Although there may be several superpixels with the similar color to s_i, only the one with the correct depth is consistent among all views. Thus, we resolve the ambiguity by voting on depth

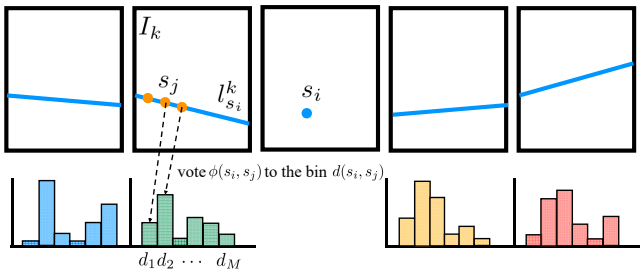

Fig. 5: The voting process for depth estimation.

values. We quantize the depth into M levels, $d_1, d_2, \ldots, d_M$. For each neighboring view I_k, we have a vote box $V^k_{s_i}$ for s_i. Each superpixel s_j locating on the epipolar line $l^k_{s_i}$ casts the vote $\phi(s_i, s_j)$ into the bin $d(s_i, s_j)$, the depth value estimated by triangulating x_i and x_j. Each bin only keeps the vote with the maximal value, i.e.,

$$V^k_{s_i}(d_m) = \max_{s_j, d(s_i,s_j)=d_m} \phi(s_i, s_j). \tag{8}$$

Figure 5 illustrates the voting process. For each hypothesized depth d_m, we obtain its probability to be the correct depth, $P_{s_i}(d_m)$, by integrating vote boxes of all neighboring views.

$$P_{s_i}(d_m) = \prod_k (\alpha + V^k_{s_i}(d_m)). \tag{9}$$

The constant α is added to avoid the malicious low probability due to occlusion. We define the set of epipolar neighbors $\mathbf{N}^e_{s_i}$ of the superpixel s_i as the superpixels locating on its epipolar lines on s_i's neighboring views. We define the cost $c(s_i, s_j) = P_{s_i}(d(s_i, s_j))\phi(s_i, s_j)$ and the epipolar color coherence term as

$$\mathbf{E}_e(s_i, s_j) = \begin{cases} c(s_i, s_j) & \text{if } s_j \in \mathbf{N}^e_{s_i}, \Phi(s_i) \neq \Phi(s_j) \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

The bottom row of Figure 4 shows the labeling coherence with this term. The false matches are successfully culled because of depth inconsistency. By incorporating the depth probability, we can better locate the corresponding superpixels and enforce the label coherence correctly.

3D correspondence consistency $\mathbf{E}_{3D}$. As described in Section 2.1, we have obtained a sparse point cloud $\mathbf{Q}$ using structure from motion. For each point $q_k \in \mathbf{Q}$, we project it onto each image and find the superpixel the projected point locates. We denote the set of superpixels corresponding to q_k as $\mathbf{N}^{3D}_{q_k}$. All superpixels in $\mathbf{N}^{3D}_{q_k}$ should be assigned with the same label since they belong to the same 3D point. Thus, we define the 3D correspondence consistency term as:

$$\mathbf{E}_{3D}(s_i, s_j) = \begin{cases} 1 & \text{if } s_i, s_j \in \mathbf{N}^{3D}_{q_k}, \Phi(s_i) \neq \Phi(s_j) \\ 0 & \text{otherwise.} \end{cases} \tag{11}$$

Both this term and the epipolar color coherence ensure consistent labeling for superpixels coming from the same 3D region.

(a) image (b) before VH (c) after VH

Fig. 6: The visual hull effectively culls the incorrect labels.

However, they are complementary to each other. The epipolar color coherence term can be applied densely to all superpixels while the 3D correspondence consistency term can only be used by the superpixels corresponding to points in the sparse point cloud $\mathbf{Q}$. On the other hand, the epipolar color coherence term is less reliable because of rough depth estimation. The 3D correspondence consistency term is more dependable since it is derived from more reliable feature matching.

2.3. Optimization

The final energy function is as the following:

$$\mathbf{E}(\Phi) = \sum_{s_i \in \mathbf{S}} \Big[\mathbf{E}_c(s_i) + \lambda_s \sum_{s_j \in \mathbf{N}^s_{s_i}} \mathbf{E}_s(s_i, s_j) + \lambda_a \sum_{s_j \in \mathbf{N}^a_{s_i}} \mathbf{E}_a(s_i, s_j) + \lambda_e \sum_{s_j \in \mathbf{N}^e_{s_i}} \mathbf{E}_e(s_i, s_j) \Big] + \lambda_{3D} \sum_{q_k \in \mathbf{Q}} \sum_{s_i, s_j \in \mathbf{N}^{3D}_{q_k}} \mathbf{E}_{3D}(s_i, s_j). \tag{12}$$

The optimization problem, $\min_\Phi \mathbf{E}(\Phi)$, is a binary labeling problem and can be solved efficiently using the s-t min cut algorithm [13]. The resultant Φ provides segmentation results.

2.4. Visual hull

The last step is to reconstruct a visual hull (VH) using the segmentation results. For constructing the visual hull, we start with a grid of voxels. For each voxel, we project it onto each view and check whether it is within the foreground region. If it locates in the background region in any view, the voxel is nullified. All remaining voxels form a volumetric model of the object and a surface reconstruction method is used for obtaining a 3D model as the visual hull. By projecting the visual hull back to each image, we remove inconsistent labeling and obtain a refined segmentation result. Figure 6 demonstrates the effectiveness of using the visual hull step. The inconsistent labels around dinosaur's legs and the top right corner in Figure 6(b) can be corrected by segmentation results of other images through help of the visual hull. After visual hull culling, we can refine the color models and go through the whole process again. The process converges if the visual hull does not deviate much from the one in the previous round.

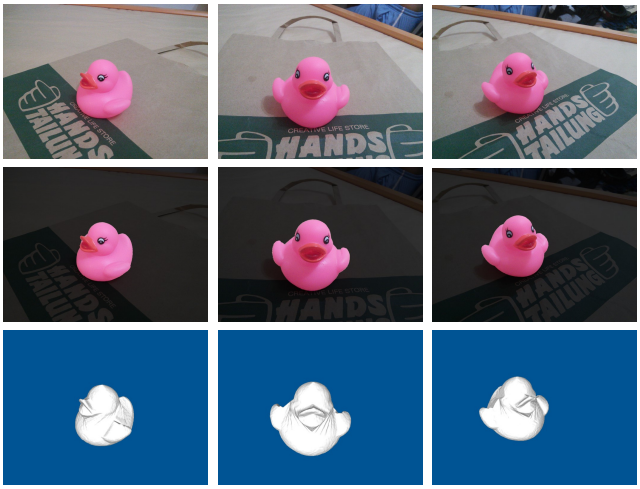

Fig. 7: The example of capturing a pink rubber duck.

3. EXPERIMENTS

We used the following parameters for all experiments: the number of Gaussians $K=10$ in Equation (1), $\gamma=10$ in Equation (5), $K_s=4$ for the appearance neighbors $\mathbf{N}_{s_i}^a$ in the appearance coherence term, the number of depth levels $M=40$, $\lambda_s=1$, $\lambda_a=1$, $\lambda_e=0.5$, $\lambda_{3D}=0.2$ in Equation (12).

The input images were captured using an App developed by us on a Google Nexus 7. The resolution of input images is 2592×1944. SLIC generates roughly 2,000 superpixels for each image. The construction of the energy terms and the graph takes around one minute for each image. The min-cut optimization takes 2 seconds. The construction of visual hull takes around one minute. The process usually converges in less than 10 iterations. Note that, among all energy terms, only $\mathbf{E}_c$ needs to be updated after each iteration. Other terms remain the same and only need to be calculated once. Thus, the update of the graph for each iteration takes little time. The optimization can be made more efficient by taking advantage of this using the algorithm by Kohli and Torr [14]. As an example, it took around 40 minutes for the segmentation of a 24-image set. The computation was performed on a notebook.

Figure 1 gives an example of capturing a toy dinosaur. We took 24 images for all examples; neighboring views are 15° apart. Figure 1(a) shows four of them. With the method described in Section 2, we obtain the segmentation results in Figure 1(b) without any user input. Figure 1(c) shows the visual hull from the corresponding views. After mapping the input images onto the surface of the visual hull, we have a textured 3D model of the toy dinosaur as shown in Figure 1(d). Note that, although we show the 3D model from the visual hull algorithm, other 3D reconstruction methods also benefit from the more accurate segmentation results of our method. A state-of-the-art 3D reconstruction pipeline is to use PMVS [3] for extracting a point cloud and then employ Poisson surface reconstruction [4] for obtaining the 3D model. Figure 1(e) shows the point cloud from PMVS. Because it is not dense enough, the reconstructed 3D model (Figure 1(f)) has many problems including fragments, holes and unwanted parts. Using better reconstruction methods such as screened Poisson surface reconstruction [5] does not help much.

Fig. 8: The example of capturing a panda doll.

Figure 7 and Figure 8 give examples of capturing a pink rubber duck and a panda doll respectively. For each example, from the top to the bottom, we show input images, segmentation results and the 3D model from the visual hull.

Figure 9 captures a sunscreen tube. There are 24 images (Figure 9(a)). Our method gives reasonable segmentation results (Figure 9(b)) and the visual hull model (Figure 9(c)) looks good. Again, PMVS does not generates a sufficient number of points (Figure 9(d)) since there is not much texture on the tube and most visual correspondences are not reliable. There are many problems with the reconstructed 3D model (Figure 9(e)).

Figure 10 compares our method with Autodesk 123D Catch [6], a system for generating a 3D model from images, on the dinosaur and duck examples. Similar to PMVS, 123D Catch also includes the unwanted desktop as part of the model. In addition, as shown in the insets, our method produces more detailed models on the toy dinosaur's front leg and mouth, and the mouth and the head of the rubber duck.

4. CONCLUSIONS

We have presented an automatic foreground object segmentation method for acquiring a 3D model from a set of images for 3D printing. It allows users to obtain 3D models more easily and robustly. In the future, we plan to use more advanced 3D reconstruction methods such as volumetric graph cut to obtain better models using the segmentation results.

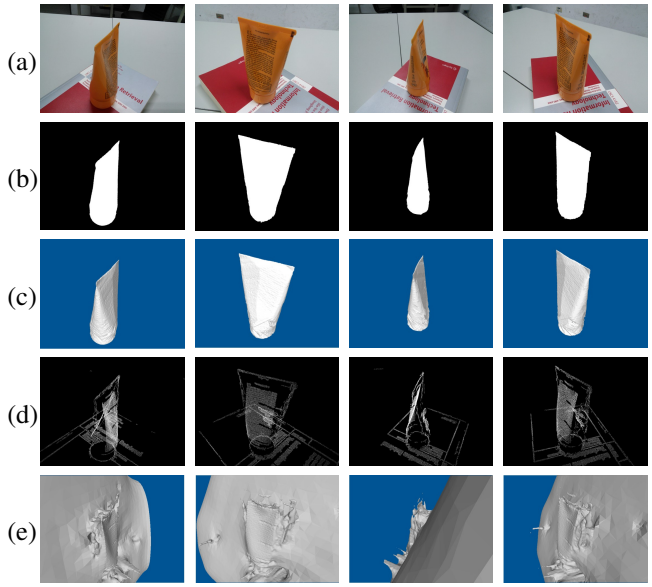

Fig. 9: The example of capturing a sunscreen tube. We compare our method with PMVS in this example.

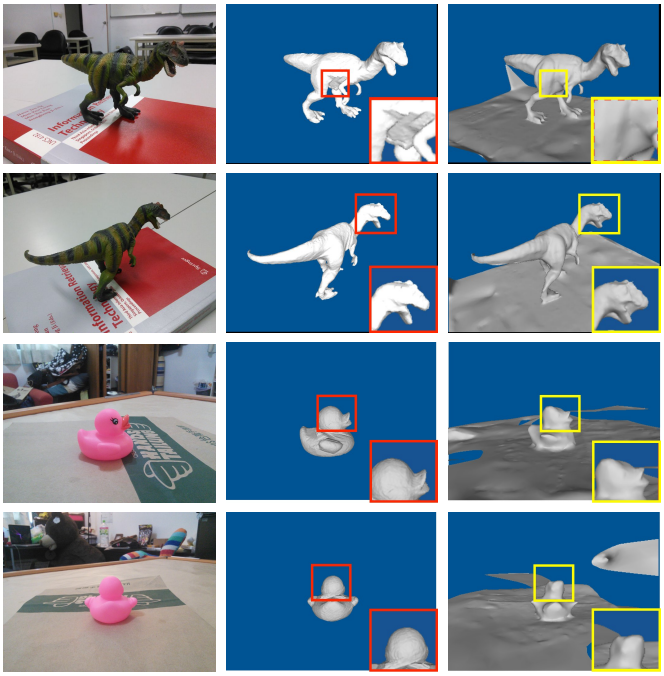

Fig. 10: Comparisons with Autodesk 123D Catch on the dinosaur and duck examples.

5. REFERENCES

[1] Simon Fuhrmann, Fabian Langguth, and Michael Goesele, "MVE - A Multi-View Reconstruction Environment," in *Proceedings of the Eurographics Workshop on Graphics and Cultural Heritage (GCH)*, 2014.

[2] Changchang Wu, "VisualSFM : A Visual Structure from Motion System," http://ccwu.me/vsfm/.

[3] Y. Furukawa and J. Ponce, "Accurate, dense, and robust multiview stereopsis," *IEEE PAMI*, vol. 32, no. 8, pp. 1362–1376, Aug. 2010.

[4] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe, "Poisson surface reconstruction," in *Proceedings of Eurographics Symposium on Geometry Processing*, 2006, pp. 61–70.

[5] Michael Kazhdan and Hugues Hoppe, "Screened poisson surface reconstruction," *ACM TOG*, vol. 32, no. 3, pp. 29:1–29:13, Jul. 2013.

[6] "AutoDesk 123D Catch," http://www.123dapp.com/.

[7] Peter Ondrúška, Pushmeet Kohli, and Shahram Izadi, "Mobilefusion: Real-time volumetric surface reconstruction and dense tracking on mobile phones," in *Proceedings of IEEE ISMAR*, Sep. 2015.

[8] Yin Li, Jian Sun, Chi-Keung Tang, and Heung-Yeung Shum, "Lazy snapping," *ACM TOG*, vol. 23, no. 3, pp. 303–308, Aug. 2004.

[9] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake, "GrabCut: Interactive foreground extraction using iterated graph cuts," *ACM TOG*, vol. 23, no. 3, pp. 309–314, Aug. 2004.

[10] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Susstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE PAMI*, vol. 34, no. 11, pp. 2274–2282, Nov. 2012.

[11] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, second edition, 2004.

[12] N.D.F. Campbell, G. Vogiatzis, C. Hernandez, and R. Cipolla, "Automatic object segmentation from calibrated images," in *Proceedings of Conference for Visual Media Production (CVMP)*, Nov. 2011, pp. 126–137.

[13] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision," *IEEE PAMI*, vol. 26, no. 9, pp. 1124–1137, Sep. 2004.

[14] P. Kohli and P.H.S. Torr, "Efficiently solving dynamic Markov random fields using graph cuts," in *Proceedings of IEEE ICCV*, Oct. 2005, vol. 2, pp. 922–929 Vol. 2.