Torres Gomez, A., & Mayol-Cuevas, W. W. (2014). Recognition and Reconstruction of Transparent Objects for Augmented Reality. In 2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR 2014): Proceedings of a meeting held 10-12 September 2014, Munich, Germany. (pp. 129-134). Institute of Electrical and Electronics Engineers (IEEE). DOI: 10.1109/ISMAR.2014.6948418

Recognition and Reconstruction of Transparent Objects for Augmented Reality

Alan Torres-Gómez* Walterio Mayol-Cuevas^
University of Bristol

ABSTRACT

Dealing with real transparent objects for AR is challenging due to their lack of texture and visual features as well as the drastic changes in appearance as the background, illumination and camera pose change. The few existing methods for glass object detection usually require a carefully controlled environment, specialized illumination hardware or ignore information from different viewpoints.

In his work, we explore the use of a learning approach for classifying transparent objects from multiple images with the aim of both discovering such objects and building a 3D reconstruction to support convincing augmentations. We extract, classify and group small image patches using a fast graph-based segmentation and employ a probabilistic formulation for aggregating spatially consistent glass regions. We demonstrate our approach via analysis of the performance of glass region detection and example 3D reconstructions that allow virtual objects to interact with them.

KEYWORDS: GLASS DETECTION, 3D RECONSTRUCTION, COMPUTER VISION, AUGMENTED REALITY.

Index Terms: H.5.1 [*Artificial, augmented, and virtual realities*]; I.4.6 [*Edge and feature detection, Region growing, partitioning*]

1 INTRODUCTION

Transparent objects are everywhere, however, due to their nature, they produce a number of phenomena that make them challenging to work with. Glass and other similar materials generally contain no easily detectable conventional Computer Vision features.

Even when there has been some work related to the detection of transparent objects, geometry extraction seems to not have been attempted, at least not without the use of additional sensors and specialist lighting. We scope our work to the analysis and reconstruction of solid, texture-less and transparent objects, mainly made of glass or rigid plastic. Our proposed method uses a conventional RGB camera, assumes no shape prior or other symmetrical constraints on the objects to be recognized and is composed of two main parts:

- *Segmentation of transparent regions:* a learning and classification-based approach where the image is broken into smaller regions for analysis, looking for cues and distortions expected to be seen on transparent objects (section 3).
- *Reconstruction from multiple views:* using previously segmented regions we use an approach similar to probabilistic space-carving to generate a point cloud containing the object's geometrical information (section 4).

* Alan.TorresGomez@bristol.ac.uk
^ wmayol@cs.bris.ac.uk

Figure 1: Recognition and reconstruction of a transparent object from an image sequence and its application into Augmented Reality. *Top left*: A set of images containing a transparent object, each from a different view point. *Bottom left:* Transparent regions are computed and used to generate a 3D model. *Right*: The augmented scene after rendering glass-like distortions and other optical effects.

While our work concerns with a specific yet relatively unexplored area in AR which focuses on the effects of real glass on virtual objects, it can also be seen as effort toward the wider problem of material-specific augmentations which has received little attention. Figure 1 shows an example result while Figure 2 shows the overall pipeline developed.

This paper is organized as follows. In Section 2 we discuss related work on transparent material detection. In section 3 we discuss our glass segmentation method. Section 4 describes the use of multiple views for reconstruction before presenting some results and description of experiments in the Sections 5 and 6. We conclude and offer directions for future work in Section 7.

2 RELATED WORK

There has been a limited amount of work on detecting transparent materials by visual means.

McHenry et al. [3] approach glass detection by analyzing the distortions introduced by a transparent objects into the background of the scene. They use the Canny [4] edge detector and analyze the edges extracted to find the distortions that would be expected to be observed in the presence of a transparent object. This is done taking samples on either side of those edges and measuring the changes in color, saturation, blurriness, texture, alpha and emission and specular reflections and feeding those measurements into a binary classifier to determine whether the edge should be labeled as "glass". Finally an active-contour model [5] is fitted around those edges labeled as glass and the area enclosed is segmented out.

A similar approach is proposed by McHenry and Ponce [6] but instead of using canny edges, they perform a segmentation of the image using the algorithm in [7]. Then, they use the classification method in [3] to find whether any two regions correspond to the same piece of background, one of them being overlaid by a transparent object and define an *affinity* measure to indicate how well two regions correspond to the same object by analyzing the junctions on the segmentation output. A geodesic active contour is used to optimize the search and find the solution efficiently.

Kompella and Sturm [8] perform transparent material detection from a single image using a collective reward based approach from a robot-mounted camera. They first define a hypothetical region where a set of points are to be found both inside and outside of it. Then, a set of support-fitness functions are evaluated on point pairs to find areas where a stronger correspondence exist. Areas that contain transparent objects are used for obstacle avoidance.

There has been additional work on finding transparent regions on images using more specialized hardware and controlled setups; those works include the use of laser range finders and stereo (Lei et al. [9], Phillips et al. [10]), depth sensors (Klank et al. [11], Lysenkov et al. [12]), and structured and polarized light (Morris and Kutulakos [13], Miyazaki et al. [14] [15]).

Fritz et al. [16] base their method on a combination of SIFT [17] and Latent Dirichlet Allocation (LDA) [18] to generate an array of transparent local features on a transparent object. Even when this provides an interesting method for object recognition, they don't provide evidence of its suitability for material segmentation or reconstruction in Augmented Reality.

Another relevant piece of work from human Vision Research and Psychology (e.g.[1], [18]) indicates that people do not necessarily have a good model of what distortions to expect from arbitrary glass objects, which can be exploited to a degree in our advantage since the exact shape and smoothness of the detected objects may not be perfectly recovered.

2.1 Our approach

Our goal is to generate 3D models of transparent objects without specialist lighting or sensors. And to the best of our knowledge, there are no other methods that address this problem from images to augmentations. Our method builds on prior work such as [6], that uses a classifier and region connectivity requiring a geodesic active contour for segmentation, and [3] which uses edge classification and also relies on active contour for segmentation. We however, base our process on small region classification and consensus voting for segmentation, which is fast and benefits from combining both edge and region information. We also employ a probabilistic space carving that exploits glass region confidence to produce the final reconstructions.

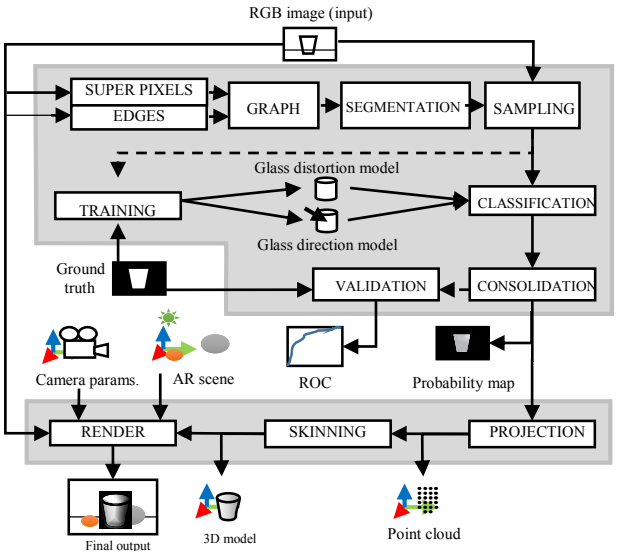

Figure 2: Segmentation and reconstruction pipeline.

3 Glass Segmentation

The appearance of any transparent object depends greatly on that of the background texture, ambient illumination, camera pose and other variables. In this section, we describe our approach to analyze an input image to find regions where it is likely to find a transparent object. This analysis is done on an automatically segmented version of the input, where measurements are taken looking for distortions that characterize a transparent object. The outline of the segmentation is based on principles described in [6].

3.1 Edge detection

One cue that stays relatively constant among changes in background is the object's outline, and it's an indication of possible boundaries between objects. But note that while edge information when combined with other priors could lead to a 3D model, it is not sufficient on its own since the determination of glass material still needs to be established. To extract the edge image $\boldsymbol{I}_E$ from the color image $\boldsymbol{I}_{RGB}$, we follow the method described on [19]. We process edges for 3 scales of the original image to preserve weaker edges.

3.2 Gradient-enhanced graph segmentation

We want to find areas of the image that belong to the same object so the whole group can be analyzed together. Recall that the aim is to segment those regions that belong to a transparent object; this makes using color a poor choice because the appearance of the glass is expected to be similar to that of the background. We use the edges extracted in the previous section as a cue for this grouping, in a way that two regions divided by an edge will produce different segments even if having similar colors. This also helps reducing the sensitivity to illumination gradients. Figure 3 shows the benefits of incorporating edge information.

The input image is first segmented into super-pixels using SLIC [20] as it provides a reasonably good initial segmentation and is faster than similar approaches. Super-pixels help to reduce the amount of information to be processed by the system in latter stages of the algorithm. Let $\boldsymbol{s}$ be the output of the super-pixel segmentation algorithm, a set containing all the super-pixels s_i of the input image. Let $B(r_1, r_2)$ be the set of pixels p at the border between any two adjacent regions r_1, r_2 in the image and is defined according to equation (1).

$$B(r_1, r_2) = \{p_{x,y} \mid p \in r_1 \bigwedge (p_{x-1,y} \in r_2 \bigvee p_{x,y-1} \in r_2)\} \quad (1)$$

Where the operator $\wedge$ stands for *logical and*, and $\vee$ for *logical or*. Note that although (1) is asymmetric, there is no significant difference between $B(\mathrm{a}, \mathrm{b})$ and $B(\mathrm{b}, \mathrm{a})$ as the change only

Figure 3: Incorporating edges on the graph segmentation. *Left*: Original image. *Center*: standard super-pixel segmentation. *Right:* our edge-enhanced graph segmentation produces less noise, less spill (note base of glass) and overall better outline of the objects.

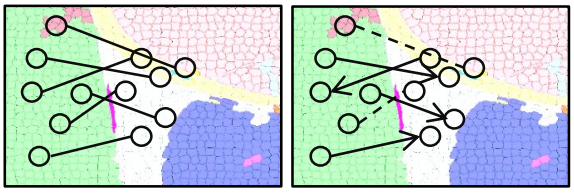

Figure 6: Sampling the segments S_i. Each sample α_i is randomly selected. *Left*: An illustration of a set of 6 samples. *Right*: After classification, each sample is associated with a glass-class and a direction pointing to where glass is expected to be.

represents an offset of a single pixel on the boundary of the two regions. The undirected weighted graph $\boldsymbol{E} = (V, E)$ is defined as the set of vertices V with a vertex defined on each super-pixel s_i of the image. The adjacent vertices are connected with an edge $E = \{ (s_i, s_j) \mid |B(s_i, s_j)| > T_{edge} \}$. The weight of an edge is defined as $w(E) = w(s_i, s_j) = \sum_{\mathrm{Pxy} \epsilon B(s_i, s_j)} \boldsymbol{I}_E(p)/|B(s_1, s_2)|$, the average magnitude along the border B over the edge image $\boldsymbol{I}_E$. Finally, the graph $\boldsymbol{G}$ is used to feed a standard graph-based segmentation algorithm [7]. The output of the segmentation is a set of segments denoted $\boldsymbol{S}$.

3.3 Image sampling and distortion measurement

Once having the image segmented into regions, the next task is to find which segments S_i correspond to a transparent object. This is done by analyzing cues in the image that indicate the presence of distortions likely to be introduced by a glass-like object.

Having the image segmented into super-pixels s and groups of super-pixels $\boldsymbol{S}$, we are now interested to know the likelihood of any segment S_i to be a component of a transparent object. In order to determine the likelihood of transparency, a subset of the super-pixels in S_i are sampled and measured to find how distorted is that group relative to its neighbors. Figure 6 shows the process of sampling on adjacent regions and votes (arrows) pointing to estimated glass locations. The samples are taken randomly as it helps to analyze each region S_i more homogeneously across multiple frames.

Let $\boldsymbol{\beta}$ be the set of all the samples taken within two adjacent segments S_i, S_j and α_i be a single instance in $\boldsymbol{\beta}$. Each sample extracted from the image is defined according to:

$$\alpha(S_i, S_j) = (s_i, s_j) \mid s_i \in S_i \bigwedge s_j \in S_j \bigwedge |B(S_i, S_j)| > 0. \qquad (2)$$

The feature vector $d_i(\alpha) = [Hue,\ Saturation, LBP, Blur,\ Alpha,\ Emission]$, is then extracted out of each sample α_i containing the numerical values of the measurements described on section 3.3.1.

3.3.1 Glass cues

We based our analysis to measure the distortions in the regions sampled on the experiments described in [3]. To compute Hue and $Saturation$ we use the distance between histograms in the Hue and Saturation channels using the HSV color-space. Changes in texture are measured as the distance on a Local Binary Patterns [21] histogram as the LBP feature. The $Blur$, $Alpha$ and $Emission$ cues are measured as described by the authors. We don't use highlights as they tend to be segmented into separated regions earlier in this process. Performance didn't suffer for its removal.

3.4 Distortion classification

To perform the sample classification we use random forests. We are interested to know how likely it is for the distortion measured to be introduced by a transparent object *(glass distortion)* and which of the two segments on that sample correspond to the transparent region *(glass direction)*.

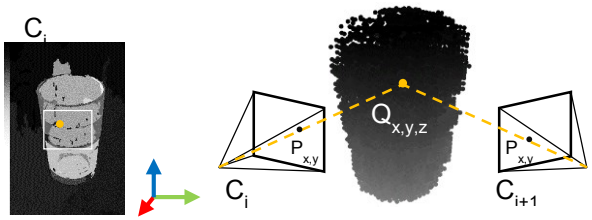

Figure 5: Each point in the point-cloud is mapped into a value in the glass intensity mask as described on section 4.1.

Two classifiers C are trained as described on 5.1. One of them, C_G is trained to find the *glass distortion* measure, a measure of the likelihood for that sample to belong to a glass region; the second, C_D is trained to determine the glass direction, this is, which region in $\alpha(S_i, S_j)$ is the one made of glass. The classifiers are fed with the feature vector d of each sample α_i taken on the sampling process described previously. The classifiers output a value k_G and k_D respectively, $0 < k_G, k_D < 1$ and proportional to the confidence the classifier has over the input data.

3.5 Consolidation

Once we compute the values g_G and g_D, we are interested in determining the likelihood of an entire region to contain a transparent object in the context of its neighbors. For any two adjacent segments S_i, S_j in the image, there is an interface vector R defined as $R(S_i, S_j) = [g_G \quad g_D]$ with $g_G = \sum_i k_{G_G}/|\beta|$ and $g_D = \sum_i k_{D_i}/|\beta|$ with $\alpha \in \beta(S_i, S_j)$.

The length in pixels of the interface between regions S_1, S_2 is $B(S_1, S_2)$ and the perimeter's length of any segment is:

$$D(S_i) = \sum_j B(S_i, S_j) \qquad (3)$$

Finally, the likelihood of any region S_1 to be glass is given by $P(S_i) = \sum_j B(S_i, S_j)\ g_G\ g_D / D(S_i)$. The glass intensity mask (I_{GI}) is then generated by mapping each segment S_i to its corresponding likelihood intensity.

3.6 Failure cases and outlier reduction

The underlying visual complexity of determining transparency leads to cases when non-transparent objects might be incorrectly detected by the algorithm. This effect is mainly produced by the local nature of the classification process. When a region in the image presents a distortion consistent to those found in glass, it becomes possible for that region to be classified erroneously. To minimize the amount of outliers detected by the algorithm, we use the additional information available by adding the multiple views of the scene. This has the value that the accuracy of the classifiers need not be as high as if this was a single frame detection problem and therefore the multiple views help to both remove outliers via lack of support as well as to allow for the 3D model to be produced. Figure 4 shows an example case where an object leads to a false positive response due to specularities and reflections on one view but not from others.

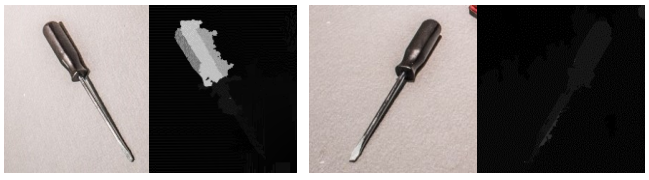

Figure 4: An object producing a false positive detection. Changing camera viewpoint helps reduce wrong detections as environmental factors (e.g. light) produce a different effect from different views.

3.7 Performance

In this paper we concentrate on the devising of the method for glass detection and reconstruction, and leave further optimizations via parallelization either on CPU or GPU out of its remit. However, in this section, we provide an analysis of the computational demands of the algorithm. We divide the analysis in two parts: the pre-processing part before classification and second, the core of our glass detection algorithm. The times reported were based on the performance of an Intel Core i7 CPU running at 3.1GHz on a single thread.

3.7.1 Preprocessing performance

There are 2 main steps on the preprocessing stage: s*uper-pixel segmentation* (see section 3.2) and *edge detection* (see section 0). For super-pixel segmentation we used a standard implementation of SLIC [20] though we note there are some GPU implementations [22]. Edge detection requires the computation of 2 convolutions of a Sobel operator (vertical and horizontal) on each channel/scale of the image (RGB, 3 scales). The number of integer multiplications required can be expressed by $\sum_n^N C_h K_x K_y I_x I_y / 2^{n-1}$ where $N$is the number of scales used, C_h is the number of channels, $K_x$and K_y are the width and height of the Sobel kernel in pixels and I_x, I_y are the image width and height in pixels. Different optimization techniques are available to optimize these methods (e.g. parallelization/GPU, convolution kernel decomposition, frequency domain analysis) that can be used to optimize the performance of the convolutions, but their analyses fall out of the scope of this paper.

3.7.2 Core algorithm performance analysis

There are a number of variables that take effect on the amount of information being processed but the most important is the number of samples analyzed and classified by the system. For any two regions S_i and S_j, this number is given by $n(S_i, S_j) = \lceil \gamma \log(|S_i|+1) \log(|S_j|+1) \rceil$, $\gamma \in \mathbb{R}, \gamma > 0$ where γ is a sampling density factor (we used $\gamma = 3$). The logarithmic relationship between the region sizes and the number of samples helps avoid the exponential growth in sampling when large regions are analyzed, rarely the number of samples per region used goes above 10 for γ equal to 1.

The global number of samples analyzed on an image depends on the number of regions found on the segmentation performed on section 3.2 and their spatial distribution. We define $\boldsymbol{F} = \{(S_i, S_j) : \exists (s_k, s_l) \in \boldsymbol{G} \mid s_k \in S_i \wedge s_l \in S_j\}$ as a set containing all the segments S_i, S_j such that there exists at least an edge on the image graph $\boldsymbol{G}$ whose vertices are contained one on each segment. The total number of samples measured on any given image can be expressed by $\sum n(S_i, S_j) \mid (S_i, S_j) \in \boldsymbol{F}$.

Table 1 illustrates the algorithm profile and times involved using a non-optimized nor parallelized implementation. As can be seen, the vast majority of time is spent in preprocessing (super-pixel segmentation and edge detection) where there is good scope for speed improvements.

Resolution	3.5 Mpx		1 Mpx		0.2 Mpx	
FPR@80%TPR	13.7%		26.6%		36.5%	
Preprocessing	83%	12.19s	75%	4.50s	61%	2.17s
Segmentation	5%	0.74s	5%	0.29s	4%	0.12s
Measurement	9%	1.35s	15%	0.88s	13%	0.47s
Classification	2%	0.22s	3%	0.19s	3%	0.11s
Consolidation	1%	0.11s	2%	0.11s	19%	0.67s
Total time		14.64s		6.00s		3.56s

Table 1. Profile of time in seconds and corresponding percentage of the time spent on each stage of the algorithm for different image resolutions while maintaining 80% of true positives

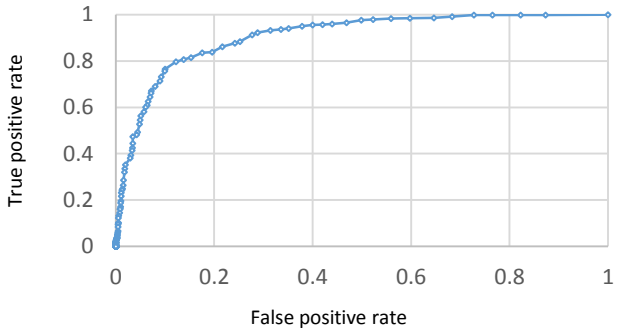

Figure 7: ROC curve on the segmentation performance computed over a set of 65 images containing different objects, background and lighting conditions manually labeled with a ground truth mask.

4 Glass reconstruction

To do the 3D reconstruction of a transparent object, we process a set of different views of the object. For our evaluation, we use a pre-calibrated camera and obtain camera poses using a calibration target for repeatability and experimental comparison purposes but there is no fundamental restriction for our work to use other camera positioning frameworks (e.g.SLAM).

The approach we take to reconstruct the object is similar in principle to probabilistic space carving [23]. Not having neither depth information nor a hard segmentation but a single intensity mask associated with each camera, we reconstruct the geometry of the transparent object by merging the glass intensity values from the segmentation into a point cloud. Sspatial support for *glassness* is sought and then removing irrelevant points by applying a threshold.

4.1 Point-cloud reconstruction and rendering

The projection is computed by setting a volume on the 3D space with points laid out regularly. Each 3D point Q_i is associated with 2 values: a counter of how many times has been observed Q_N and Q_M a floating point value corresponding to the average of intensities projected into that region of the space (see Figure 5). To extract the final point-cloud, we apply a threshold on both Q_M and Q_N, requiring that every point has been observed a minimum number of times and has a high probability of glass. A spatial filter is applied to remove outliers and the point-cloud is split into clusters, which helps separate multiple objects, if present. For each cluster, a mesh is wrapped using a convex hull. To recover cavities on the model (see cup steem in Figure 8), the mesh is then shrunk to the closest point in the cloud and softened by mesh subdivision.

In order to visualize the achievable photo-realistic rendering of transparent and augmented objects into the original scene, we load the 3D models extracted with the method and the corresponding camera parameters into Blender [24]. We use the models generated to create interactions between synthetic objects and the real scene, which are finally rendered into the original frames of the input sequence. This could in principle be integrated as well with a real-time rendering engine in an AR setting as it has been demonstrated recently by Kan et al. [25].

5 Experiments

For evaluation of the segmentation performance of the proposed method, a set of 65 non-trained, high resolution (~10Mpx) images were fed into the system. These images contain a transparent object in different configurations of background, position and illumination, including new objects never seen by the system (e.g. tinted glass). For quantitative results, a ground truth mask was manually generated for each test image. We also visually show some example results in Table 2.

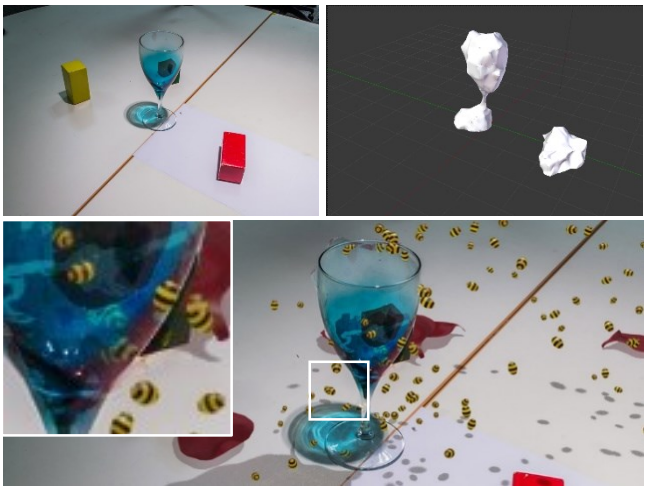

Figure 8: The reconstruction of a transparent object and its use on AR. *Top left*: A scene is capured taking pictures from multiple view-points (we used 9). The images are processed using the method described on section 3. *Top right*: The object is reconstructed as a 3D model as described on section 4. Some kinds of objects have effects similar to those of glass, which result in a false positive. *Bottom*: An augmented scene using information from the reconstruction. Virtual objects (bees) interact with the transparent object. Optical effects are added to objects behind glass (see detail and accompanying video).

5.1 Classifier training

To train the classifiers, we used a data set consisting of 200 images with a mixture of positive and negative examples. None of these images were used for testing. Training and testing sets were captured with a combination of different cameras, backgrounds and illumination conditions. From there, about 100,000 samples (see section 3.3) were extracted for training. Images had corresponding ground truth masks. Multiple clear transparent objects were used during training, none of them had tint. As the distortions introduced by glass tend to be quite subtle, we use a relatively large-sized forest (~50 trees, depth ~25). We determined our configuration by iteratively increasing the size of the forest up to the point where the benefit for each new tree was negligible. Each classifier is fed as described on section 3.4.

6 Results

We perfomed a ROC analysis to evaluate the overall classification on the individual image segmentation (see Figure 7). Rates of about 80% for true positives with 13% of false positives are achieved. This is better or comparable with the results cited by methods reported in [3] or [6] which use iterative Active Contours as additional outlining contraints for boundary determination. And while our current implementation is not yet real-time it is close (Table 1). Table 2 illustrates the response of the system. Images contain a variety of object textures, lighting conditions and different transparent objects never seen by the system. One interesting case is the correct segmentation of tinted glass objects (see Table 2-2) that were not contained on the training set. Even when color similarity is an important cue to determine transparency, other cues like texture distortion helped to produce the desired effect.

In Figure 1 and Figure 8, results of detecting and reconstructing glass are shown. A direct comparison with existing methods for transparent object reconstruction ([11], [13], [14], [15]) is not straightfoward; some of them produce very high quality reconstructions but none of these works deal with reconstruction in an uncontrolled environment without the use of specialized hardware. This limits their application in an AR setup on mobile plattforms where the only standard sensor commonly available is a simple RGB camera. Even when the models generated by our method are not perfect, they are good enough to produce convincing glass-like distortions and enable interactivity between the real transparent objects and virtual augmentations in AR. Please note that the models were generated with a small set of images captured with a mobile phone camera (Nexus 4). Results are best seen on the accompanying video material.

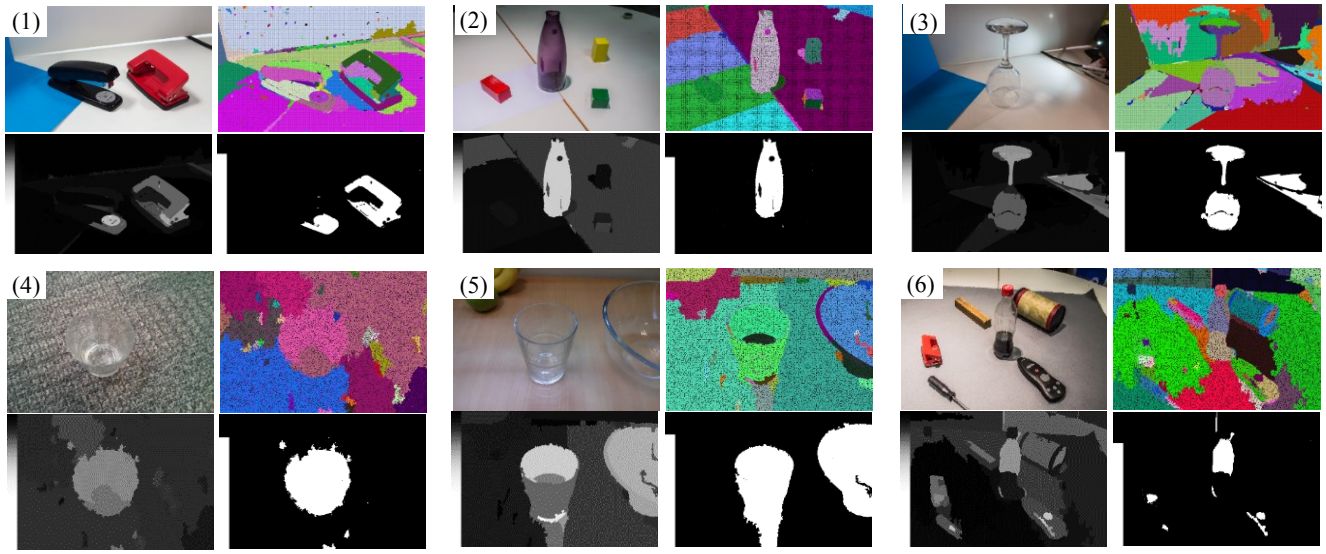

Table 2: Test cases of the glass-segmentation method described in Section 3. *Top left*: Color input image. *Top right*: The image is broken into segments. *Bottom left*: Glass intensity mask extracted after classification and consolidation of the image. Higher values (clearer colors) represent areas with higher likelihood of containing a transparent object. Contrast was artificially improved for visualization purposes. *Bottom right*: A threshold is applied to the output image. The threshold is set to allow an acceptance rate of 80% based on the ROC curve in Figure 7. Cases 2-6: Different transparent objects on multiple conditions, backgrounds, materials and captured with different cameras. Case *1:* A test case where false positives are detected on some metallic and shiny materials.

7 Conclusion

It has been shown that transparent object detection and reconstruction using the described method is feasible. This method performs an analysis of the distortions introduced by transparent objects into the scene. The results are visually convincing and are an example of a relatively unexplored area in AR.

There are some limitations of the system that need to be addressed in future work. Some of them include better handling of false positives which are usually generated when different parts of the same object introduce optical distortions that correspond to those of the trained models (see the red block in Figure 8). Some amount of clutter can be handled by the system but there is still room for improvement on more complex scenarios, specially handling objects with more textures. Occlusion handling (including self-occlusion) is other area that need to be addressed in future work.

There are certain optical effects that also produce a high response of the system, such as some shadows and reflections produced by shiny or metallic objects. Although these effects might result problematic in an AR setup, they might be as well useful for other kind of applications that might benefit from them.

Natural extensions include investigating further optimizations and the integration to real-time camera pose estimation, motion blur handling and real-time rendering of transparent objects. We also feel that further information can be obtained from temporal information as well as other optical phenomena that could be incorporated or even extracted from in-situ live training.

References

[1] R. Fleming, J. Frank and T. Laurence, "Visual perception of thick transparent materials," *Psychological science,* vol. 22, no. 6, pp. 812-820, 2011.

[2] R. Fleming, T. A. and A. E.H., "Specular reflections and the perception of shape," *Journal of Vision,* vol. 4, no. 9, 2004.

[3] K. McHenry, J. Ponce and D. Forsyth, "Finding glass," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 2005.

[4] J. Canny, "A computational approach to edge detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* no. 6, pp. 679-698, 1986.

[5] V. Caselles, R. Kimmel and G. Sapiro, "Geodesic active contours," *International journal of computer vision,* v. 22, n. 1, pp. 61-79, 1997.

[6] K. McHenry and J. Ponce, "A geodesic active contour framework for finding glass," in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, 2006.

[7] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *International Journal of Computer Vision,* vol. 59, no. 2, pp. 167-181, 2004.

[8] V. R. Kompella and P. Sturm, "Collective-reward based approach for detection of semi-transparent objects in single images," *Computer Vision and Image Understanding,* vol. 116, no. 4, pp. 484-499, 2012.

[9] Z. Lei, K. Ohno, M. Tsubota, E. Takeuchi and S. Tadokoro, "Transparent object detection using color image and laser reflectance image for mobile manipulator," in *Robotics and Biomimetics, 2011 IEEE International Conference on*, 2011.

[10] C. J. Phillips, K. G. Derpanis and K. Daniilidis, "A novel stereoscopic cue for figure-ground segregation of semi-transparent objects," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, 2011.

[11] U. Klank, D. Carton and M. Beetz, "Transparent object detection and reconstruction on a mobile platform," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, 2011.

[12] I. Lysenkov, V. Eruhimov and G. Bradski, "Recognition and Pose Estimation of Rigid Transparent Objects with a Kinect Sensor".

[13] N. J. Morris and K. N. Kutulakos, "Reconstructing the surface of inhomogeneous transparent scenes by scatter-trace photography," in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, 2007.

[14] D. Miyazaki, M. Kagesawa and K. Ikeuchi, "Transparent surface modeling from a pair of polarization images," *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. 26, no. 1, pp. 73-82, 2004.

[15] D. Miyazaki, N. Takashima, A. Yoshida, E. Harashima and K. Ikeuchi, "Polarization-based Shape Estimation of Transparent Objects by Using Raytracing and PLZT Camera, in Proceedings of SPIE (Polarization Science and Remote Sensing II, Part of SPIE's International Symposium on Optics and Photonics 2005)," 2005.

[16] M. Fritz, M. Black, G. Bradski and T. Darrell, "An additive latent feature model for transparent object recognition," *NIPS,* 2009.

[17] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision,* vol. 60, no. 2, pp. 91-110, 2004.

[18] D. M. Blei, A. Y. Ng and M. I. Jordan, "Latent dirichlet allocation," *the Journal of machine Learning research,* v. 3, pp. 993-1022, 2003.

[19] S. Hinterstoisser, C. Cagniart, S. Ilic, P. Sturm, N. Navab, P. Fua and V. Lepetit, "Gradient response maps for real-time detection of textureless objects," *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. 34, no. 5, pp. 876-888, 2012.

[20] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua and S. Susstrunk, "Slic superpixels," *Ecole Polytechnique Federal de Lausssanne (EPFL), Tech. Rep,* vol. 149300, 2010.

[21] T. Ojala, M. Pietikainen and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* vol. 24, no. 7, pp. 971-987, 2002.

[22] C. Y. Ren and I. Reid, "gSLIC: a real-time implementation of SLIC superpixel segmentation," 2011.

[23] A. Broadhurst, T. W. Drummond and R. Cipolla, "A probabilistic framework for space carving," in *ICCV*, 2001.

[24] "www.blender.org," [Online].

[25] P. Kan and H. Kaufmann, "High-quality reflections, refractions, and caustics in augmented reality and their contribution to visual coherence," in *Mixed and Augmented Reality (ISMAR), 2012 IEEE International Symposium on*, 2012.

[26] C. Rother, V. Kolmogorov and A. Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," in *ACM Transactions on Graphics (TOG)*, 2004.

[27] A. Hofhauser, C. Steger and N. Navab, "Perspective planar shape matching," in *IS SPIE Electronic Imaging*, 2009.

[28] Netz and Osadchy, "Recognition using Specular Highlights," 2012.

[29] E. H. Adelson and P. Anandan, Ordinal characteristics of transparency, Vision and Modeling Group, Media Laboratory, Massachusetts Institute of Technology, 1990.