2016 IEEE International Symposium on Mixed and Augmented Reality

PPV: Pixel-Point-Volume Segmentation for Object Referencing in Collaborative Augmented Reality

Kuo-Chin Lien* Benjamin Nuernberger† Tobias Höllerer‡ Matthew Turk§

University of California, Santa Barbara

ABSTRACT

We present a method for collaborative augmented reality (AR) that enables users from different viewpoints to interpret object references specified via 2D on-screen circling gestures. Based on a user's 2D drawing annotation, the method segments out the user-selected object using an incomplete or imperfect scene model and the color image from the drawing viewpoint. Specifically, we propose a novel segmentation algorithm that utilizes both 2D and 3D scene cues, structured into a three-layer graph of pixels, 3D points, and volumes (supervoxels), solved via standard graph cut algorithms. This segmentation enables an appropriate rendering of the user's 2D annotation from other viewpoints in 3D augmented reality. Results demonstrate the superiority of the proposed method over existing methods.

Index Terms: H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Artificial, augmented, and virtual realities;

1 INTRODUCTION

Identifying objects in a 3D environment is an important step in augmented reality (AR) applications. In AR, virtual content must be appropriately registered with the physical scene, and annotations often refer to more than just a single 3D point. Annotations can be made to an object in the physical scene or to a part of such an object, and hence, such "spatially dependent components" of AR annotations [28] need to be semantically identified.

One well-known simple example is identifying the ground plane on which virtual characters in an AR game can interact [12]. Another example would be AR virtual characters finding a believable place to sit, based on recognition of physical chairs in the real scene [22].

In many scenarios, however, techniques such as plane detection and object recognition are not sufficient if the physical object of interest cannot be determined until the user specifies it during interaction. For example, the user may be interested in a specific 3D surface in the physical scene or an object that is not known by the system a priori. This scenario is particularly relevant in multi-user, interactive AR labeling applications such as AR-based remote assistance and collaborative navigation. In these applications, the AR task quite often depends on the context, which must be interactively clarified during run time. Furthermore, the virtual annotations inserted by users can refer to objects of interest of many different sizes and shapes that may not be known beforehand (*e.g.*, a machine part to be replaced or an arbitrary landmark to guide the navigation in an unfamiliar environment).

*(co-first author) e-mail: kuochin@ece.ucsb.edu

†(co-first author) e-mail: bnuernberger@cs.ucsb.edu

‡e-mail: holl@cs.ucsb.edu

§e-mail: mturk@cs.ucsb.edu

Therefore, a key question is: how can we identify the user-specified object/part in real time to enable instant multi-party visual communication in an AR collaboration application? This is particularly challenging because the users can explore a shared reconstructed scene yet may have independent 2D views of the scene [23], thus causing challenges for 3D referencing techniques in collaboration scenarios [19].

Based on recent studies [4, 11, 18], we chose to focus the scope of this paper on 3D object referencing via the assistance of a user-drawn 2D circle annotation since (1) on-screen drawing is a simple way to convey information, especially when directly accessing the object is impossible (*e.g.*, due to far distances, hazardous environments, or in remote assistance scenarios), (2) a single pointer (*e.g.*, a red dot) has been shown to provide insufficient cues compared with 2D drawing annotations [11], and (3) a circle has been shown to be the most common 2D gesture annotation for referencing an object in a scene [18]. Note that we use the term circle to loosely include any kind of closed loop 2D drawing that roughly resembles a circle, such as an ellipse [18].

In this paper, we propose a new method to locate the objects to which the user-drawn on-screen circles refer in general multi-user collaborative AR scenarios. Unlike existing techniques [4, 11, 18], our method does not require the object of interest to be planar or nicely isolated for a careful scan. Instead, our method combines the merits of both plane determination and 3D structure inferencing methods by simultaneously extracting the 2D image pixels as well as the 3D scene points associated with the object in a unified three-layer graph relating pixels, points, and volumes to each other. Our system is called "PPV" in reference to these three spatial components, but it also refers to the "paint-per-view" process of indicating annotations in separate 2D input views. Our method is designed for interactive real-time scenarios, such that it can work comfortably with a room-sized AR scene reconstructed with a modern SLAM algorithm [27].

We organize the remaining sections of the paper as follows. In Section 2, we describe the challenges of using 2D circle annotations for 3D object localization with respect to prior work in 3D AR annotation and object segmentation. Section 3 introduces the structure of the proposed three-layer graph. Based on this graph, we describe in Section 4 how an efficient, unified inference can be made to simultaneously locate the object both on the image plane and in 3D space. Our implementation is described in Section 5, followed by experiments in Section 6. Finally, discussion and future work are described in Section 7, followed by a summary in Section 8.

2 CHALLENGES AND RELATED WORK

There are two major challenges a system needs to handle when using 2D circle annotations for locating 3D objects in AR scenes: (1) 2D input which cannot inherently directly describe something in 3D without making certain inferences, and (2) a limited observation of the scene simply due to working in unprepared environments for augmented reality.

978-1-5090-3641-7/16 $31.00 © 2016 IEEE
DOI 10.1109/ISMAR.2016.21

2.1 Lower Dimensionality Input

First, 2D drawings inherently have limited representational power to describe a 3D entity due to the lower dimensionality of the object referencing input.

Consider a projection of the 2D drawn enclosure from the drawer's viewpoint to the 3D space—it can only provide a 3D cone in which the 3D object of interest may reside. In other words, a 2D circle can be seen as a projection of a 3D ring that is possible to reside *anywhere* on the cone surface. In line with this interpretation, a number of plane fitting approaches have been proposed [4, 10] to locate the 3D ring and, in some sense, the 3D object of interest equivalently. This overall approach may be likened to a more general version of flashlight selection [13].

Another popular interpretation to the 2D user-drawn annotations is graffiti/spray-paint [6, 25], which assumes the drawing trajectory is on those visible, nearest 3D points (from the drawer's viewpoint) associated to the painted pixels. It does not enforce the drawing path to form a ring in 3D space, which gives more freedom to interpret the drawing annotation in some sense. However, this method is not directly applicable to using circle annotations for referencing purposes. The reason is that, unlike pointers or scribble annotations that have also been used for referencing purposes in AR [3, 16], the path of a casually drawn circle is not necessarily on the object to be referred to (Figure 1). In a worst case, all of the trajectory points are painted on some other nearby objects (in the view of the drawer's 2D image plane), but none of them are on the object that the user intends to refer to. As a result, neither these points in 3D nor their projections to another viewpoint help achieve the goal of referencing the object of interest. Ultimately, both plane fitting and graffiti methods are only suitable for referencing a planar object or a surface of an object.

2.2 Limited Observation of the Scene

The second challenge of locating 3D objects given 2D circle annotations in AR applications is that only a limited observation of the scene may be available. In addition to the limitations of having two-dimensional drawing annotations as input, the color images on which the user can draw are also two-dimensional. Note that there exists a subtle difference between the challenges of referencing 3D objects using 2D annotations and inferring scene geometry and semantics from 2D images: a 2D annotation inherently requires disambiguation due to lower dimensionality input, while a 2D image is a limited observation of a 3D world.

To overcome these challenges, the object localization algorithm can be directly performed on the reconstructed 3D scene to take advantage of the accessible 3D geometric information; this is readily available in many situations since modern scene reconstruction algorithms [8, 17] allow an object to be efficiently scanned and modeled in just a few seconds. A number of prior works fall in this category: Tatzgern *et al*. [24] determine a ground plane in a 3D point cloud. After removing the plane points, objects can be identified by clustering the remaining points. This method is limited to simple scenes where objects are placed on the ground or on a table. Golovinskiy and Funkhouser [5] propose to allow users to specify the center and radius of an object to be interactively extracted from a 3D point cloud. Meyer and Do [15] propose a more convenient GrabCut-style [21] user interface which projects a 2D drawn bounding rectangle to be a 3D bounding box for 3D mesh segmentation. However, none of these algorithms consider any rich 2D image cues but instead only rely on the underlying 3D model for object segmentation. In addition, they overlook the drawing user's effort on recognizing objects from a sparse, and likely low quality, 3D model.

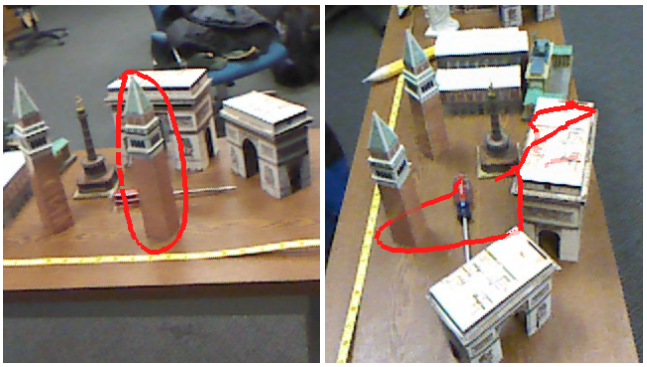
SemanticPaint [26] is another method that operates directly on the 3D scene. It segments the scene via interactive touch gestures and machine learning using a Conditional Random Field model. Their method continuously learns scene labels from the online segmentations and propagates such labels to newly seen areas of the environment accordingly. Our method does not rely on pre-defined scene knowledge such as ground plane or online learned semantics, but we see our method as complementary to such approaches (*e.g*., our method may be used as an input to guide online learning algorithms and to guide the initial segmentation).

To avoid users directly operating on a reconstructed 3D model, Nuernberger *et al*. [14, 18] propose to look for a 3D convex hull that a 2D image-plane segmentation result belongs to. However, their convex object assumption, while inspired from cognitive science research [7], is not always applicable in AR for identifying individual components on a carefully-scanned, human-made object. The problem is that when we consider a general AR task in a cluttered room, where objects can be stacked on each other or leaning on a wall instead of being nicely isolated for scanning and modeling, even a 3D convex object can appear to be a 2D plane as some surfaces of the object are never scanned, which breaks the convex hull assumption. This degeneration of sensing a 3D convex object as something that looks like a 2D plane also occurs in outdoor scenes because an object can be far away such that only a face of it can be scanned until we approach the object up close.

In summary, all these methods strongly rely on the quality/completeness of the underlying 3D model. In contrast, our proposed method jointly optimizes the extractions of the 2D pixels and the 3D scene points associated with the object in a unified graph-based framework as described next. Table 1 shows how our method, compared to previous approaches, can appropriately handle both planar and convex objects of interest using a single framework.

Annotation Methods	Planar	Convex
Spray-paint [6, 25]	✓	
Planar interpretations [4, 10]	✓	
Convex 2D-3D co-segmentation [18]		✓
Our method	✓	✓

Table 1: Annotation methods for object referencing in 3D augmented reality via 2D on-screen gestures and whether or not they appropriately can handle different types of object geometry.



(a) Original drawing viewpoint. (b) Spray-paint interpretation from a different viewpoint.

Figure 1: Illustration of a spray-paint interpretation of 2D circle annotations in 3D augmented reality used in many prior works (*e.g*., [6, 25]).

3 The Proposed Three-Layer Graph

We formulate the object localization problem as an optimization problem on a graph. We construct a graph $G = \{E, V\}$ where the

edges E describe how two vertices influence each other, and the vertices V consist of all the 2D pixels V_x in the drawing frame, 3D scene points V_y, and supervoxels V_z based on them. The optimization goal is to label each vertex in this graph to be either belonging to the target foreground object F or the background scene B, given user assigned initial values, *i.e.*, the drawn circle, on V_x. Note that while the number of 3D scene points can keep growing as the person holding the camera keeps exploring the scene in an online setting, here the graph modeling and optimization are based on the latest available 3D scene model when the drawing is made.

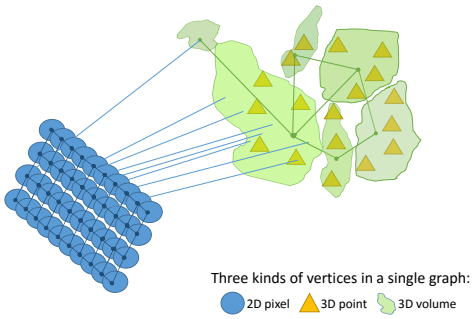


Figure 2: The proposed three-layer graph. Blue circles: image pixels; Orange triangles: 3D points; Green regions: supervoxels.

Figure 2 illustrates the structure of our three-layer graph. Three kinds of vertices are marked as three different colors: the blue circles denote pixels on the 2D image plane, each connected with four neighboring pixels; orange triangles denote 3D points associated with some structural object in the scene; green regions represent supervoxels, where each can be connected with multiple 2D and 3D vertices. The main reason for constructing these supervoxels is that we want to indirectly exchange information between the native 2D and 3D layers with some tolerance of any imperfect 2D-3D registration (a 3D point is typically generated by triangulating or fusing several 2D pixel observations from different views and is therefore not necessarily perfectly aligned with the corresponding pixel in the current image).

Specifically, we adopt Papon *et al.*'s VCCS algorithm [20] to efficiently over-segment the space in which the 3D points and the back-projected 2D pixels reside. As a result, every 3D point is covered by some supervoxel while some 2D pixels may not be associated with a supervoxel if they fall outside of the reconstructed scene. In addition, each supervoxel region is associated with at least one point, which can be a point in the scene model or a 2D pixel. We further allow message passing between adjacent supervoxels, as illustrated with the green edges connecting two supervoxel regions. Compared with other graph-based methods which construct edges directly on the 3D model [9, 15], this design enables significantly faster graph construction since the number of supervoxels (also the edges between them) is constantly hundreds of times smaller than that of 3D points in our experiments.

As can be seen, this graph nicely connects the 2D image pixels and 3D scene points, such that 2D and 3D information can be freely passed to influence and benefit from each other, as the color image typically has a higher resolution and the scene model can provide more geometric cues. Moreover, the integration of image and scene model into a single graph allows us performing an efficient, unified global optimization to solve the object localization problem, as will be detailed in the next section.

4 Unified Optimization

As briefly mentioned in the introduction, quite a few previous works also try to interpret a user's 2D drawing in 3D. However, they all perform two-stage reasoning strategies, which first either determine the object region or pre-process the drawn trajectory on the image plane and then accordingly infer the object location in 3D based on the scene model. This paper, in contrast, proposes a unified energy minimization approach to simultaneously locate the user's chosen object in the image as well as in the scene model while taking into account information from both domains. The energy function we minimize is Equation (1):

$$\Phi(G) = \Phi_x + \Phi_y + \Phi_{xx} + \Phi_{xz} + \Phi_{yz} + \Phi_{zz} \tag{1}$$

where Φ_x and Φ_y codify the color likelihood and locality probability of V_x and V_y based on the user drawn circle. Formally,

$$\Phi_x = \sum_{v \in V_x} \psi_x(v), \tag{2}$$

where $\psi_x(v) = -\log p(C_v, L_v | \alpha_v)$ determines the cost of assigning a vertex v to $\alpha_v \in \{F, B\}$ based on its color C_v and locality L_v information. Specifically, for a v inside the user drawn circle, the conditional probability p is estimated by evaluating the foreground and background color Gaussian Mixture Models (GMM) obtained by collecting pixels inside and outside the drawn circle respectively[1]; for pixels outside the circle, we assign a constant cost to penalize a pixel being labeled as foreground. Similarly,

$$\Phi_y = \sum_{v \in V_y} \psi_y(v), \tag{3}$$

where $\psi_y(v)$ yields a constant penalty for a 3D point if it is outside the 3D cone projected from the 2D circle yet is labeled as foreground. For a 3D point inside the cone, it is not clear if the point belongs to the target object or another object (*e.g.*, occluding or being occluded by the target object). Therefore, we remain neutral, *i.e.*, let $\psi_y = 0$, in this case. $\psi_y(v)$ only depends on locality since the scene point cloud can be sparse and color models built on it can be unreliable.

The remaining four terms in Equation (1) define how to pass message among vertices. Φ_{xx} is a standard edge-sensitive term in image processing [1] allowing two neighboring pixels similar in color to influence each other. Formally,

$$\Phi_{xx} = \sum_{(i,j) \in E_{xx}} \mathbb{1}_{[\alpha_i \neq \alpha_j]} (\lambda_1 + \lambda_2 e^{-\mu \|c_i - c_j\|^2}), \tag{4}$$

where E_{xx} are edges at the pixel layer, and $\mathbb{1}_{[\ldots]}$ is an indicator function equal to one if two adjacent pixels are assigned different labels. The last term scaled with λ_2 adds larger cost in addition to the base cost λ_1, if the two pixels i and j have similar color c_i and c_j. The constant parameter μ is defined to be:

$$\mu = \frac{1}{2 \sum_{(i,j) \in E_{xx}} \frac{\|c_i - c_j\|^2}{|E_{xx}|}}, \tag{5}$$

where the cardinality $|E_{xx}|$ denotes the number of edges between 2D vertices.

Φ_{zz} is where we implement the convexity constraint. We encourage two neighboring supervoxels to get the same foreground/background label:

$$\Phi_{zz} = \sum_{(i,j) \in E_{zz}} \mathbb{1}_{[\alpha_i \neq \alpha_j]} \psi_{zz}(i,j), \tag{6}$$

[1]Color GMMs are modeled only once, when a circle is drawn by the user. We did not update color models during segmentation although it could be done (*e.g.*, [21]) in future work.

where E_{zz} are edges at the supervoxel layer, and $\psi_{zz}(i,j) = W_{zz}\mathbb{1}_{[\theta_{dihedral}>(\pi/2-\theta_{tol})]}$ is a weighted indicator function evaluating if two neighboring clusters of points form an obtuse angle, *i.e.*, geometrically more likely to belong to the same object [7]. We apply a weight $W_{zz} = \frac{|V_y|}{|V_z|}$ to balance this term with the others as the number of V_z is typically much smaller than that of the native vertices.

Φ_{xz} and Φ_{yz} enable cross-layer information propagation as follows:

$$\Phi_{xz} = W_{xz} \sum_{(i,j)\in E_{xz}} \mathbb{1}_{[\alpha_i \neq \alpha_j]}, \quad (7)$$

$$\Phi_{yz} = W_{yz} \sum_{(i,j)\in E_{yz}} \mathbb{1}_{[\alpha_i \neq \alpha_j]}, \quad (8)$$

where E_{xz} and E_{yz} are cross-layer edges. The two energy terms Φ_{xz} and Φ_{yz} penalize the points associated with the same supervoxels yet receiving different labels during the optimization. W_{xz} and W_{yz} are constant weights.

With the energy terms being defined as such, one can then apply a graph cut algorithm [2] to determine an optimal labeling for V, equivalently, the location of the target object both in 2D and in 3D, considering all information provided from all layers. In this unified framework, each energy term contributes different hints from the user, the 2D, or the 3D observations for a global optimization rather than making its own hard decision and passing along it to the next stage. In the following sections, we show how well the guaranteed optimal solution outperforms prior work which can be trapped in a local minimum.

5 Implementation

We created a working prototype of our method using a Microsoft Kinect sensor and Elastic Fusion [27] for visual SLAM. Two visualizations of the referenced object were implemented: (1) a 2D ellipse fitted to the 3D segmented points projected onto the rendering plane [18]; and (2) highlighted 3D points with periodic time-varying opacity. See the right column of Figure 8 for an illustration of the highlighted 3D points visualization.

In our unoptimized implementation, the method runs in about 1.93 seconds for a point cloud of over 300,000 points on an Intel Core i7-4790 CPU with a Quadro K5000 GPU and 16GB of RAM. To avoid any interruption to the interaction and visual SLAM processing, we run our method in a background thread; a spray-paint visualization of the annotation is shown until the segmentation is completed, at which point either the 2D ellipse or highlighted 3D points are shown.

In all experiments, the following parameters were used: $\lambda_1 = 4$, $\lambda_2 = 8$, $W_{xz} = 0.5$ and $W_{yz} = 10$.

6 Experiment

We ran a qualitative experiment comparing against what we considered to be the most competitive related methods for object referencing in augmented reality via 2D on-screen gestures. Specifically, we compared against a 3D planar annotation interpretation, the median depth plane [4], and Nuernberger *et al.*'s convex 2D-3D co-segmentation algorithm [18].

In order to evaluate the methods in general environmental settings, we recorded both outside-in and inside-out scenes as datasets used for the experiments. In the outside-in datasets, we let the camera move around target objects to capture them from many perspectives; in the inside-out datasets, the layout of the scene prohibits the camera to make an orbital observation on the objects so only parts of the objects are scanned. See two sample models in bird's eye views in Figure 3.

In the evaluation, each method was judged on how well it could appropriately transfer a 2D on-screen circle gesture drawing to a novel viewpoint. Such a scenario may occur in AR collaboration where users do not share the same view, such as in co-located, remote, or asynchronous collaboration (*e.g.* [4,19,23]); we also imagine that this scenario is useful in situations where reconstructed environments are explored and semantically labeled collaboratively (*e.g.*, exploring and labeling items of interest in real estate via Matterport reconstructed scenes).

(a) Outside-in: scanning objects placed on a table.

(b) Inside-out: walking by a bookcase and scanning objects in it.

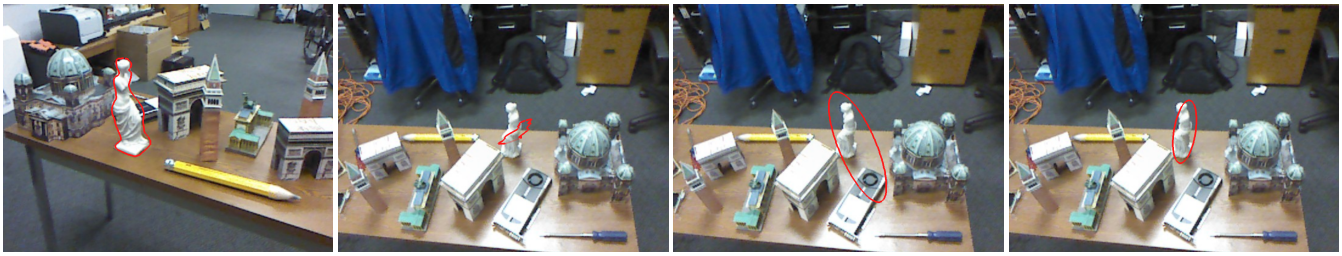
Figure 3: Two scenes modeled in the outside-in fashion and the inside-out fashion, respectively. Black frusta illustrate the camera trajectories; Magenta locates the latest position of a camera. Dark blue indicates unmodeled regions.

There were 4 objects of interest used per scene and 3 different viewpoints for each object. For outside-in scenes, we used viewpoints with approximately 0°, ±90°, and 180° differences in viewing angle with respect to the object of interest. For inside-out scenes, we used viewpoints with approximately 0°, ±45°, and ±90° differences in viewing angle with respect to the object of interest.

To gather qualitative feedback from a wide range of users, we conducted a survey using Amazon Mechanical Turk (AMT), asking participants to compare between the three methods. Specifically, similar to Nuernberger *et al.* [18], we showed the participants a set of four images for a given object referenced via a 2D on-screen gesture: (1) the original 2D on-screen gesture from the drawing viewpoint; (2-4) a new viewpoint showing each of the three methods' ellipse rendering results. Each AMT participant was asked, "Which image (left, middle, or right) best conveys the same meaning as the red drawing in the first picture?" The order of the images was randomized to avoid any bias effects. We invited only partic-

(a) Original drawing. (b) Median depth plane [4]. (c) 2D-3D co-segmentation [18]. (d) Proposed method.

Figure 4: An outside-in experiment result showing the annotation from a drastically different viewpoint. In general, 3D planar interpretations of 2D on-screen gestures are unable to handle 3D convex objects. However, our proposed method as well as convex 2D-3D co-segmentation [18] give decent results.



(a) Original drawing. (b) Median depth plane [4]. (c) 2D-3D co-segmentation [18]. (d) Proposed method.

Figure 5: An outside-in experiment result showing the annotation from a drastically different viewpoint. In general, 3D planar interpretations of 2D on-screen gestures are unable to handle 3D convex objects. In this example, our proposed method gives the best results, as verified by the AMT votes: 2 for median depth plane, 2 for 2D-3D co-segmentation, and 18 for the proposed method.

ipants in the United States who have good records in AMT user studies (*i.e.*, $> 90\%$ hit approval rate) to ensure that our questions are clearly understood and carefully answered.

40 users participated in the study, 20 for the OutIn_1 & InOut_1 datasets (Figure 3a & Figure 3b) and 20 for the OutIn_2 & InOut_2 datasets. Results are summarized in Table 2 and shown in Figures 4 through 7. Figure 4 shows a case where the convex assumption by Nuernberger *et al*. [18] works well and gives similar results to our method. Figures 4 and 5 illustrate how the median depth plane method [4] does not adequately transfer the annotation to new viewpoints. Figure 7 illustrates a case where the convexity assumption of Nuernberger *et al*. [18] causes an over-segmentation, whereas our proposed method better conveys the meaning of the original drawing.

Method	OutIn_1	OutIn_2	InOut_1	InOut_2
Median depth [4]	20	36	39	25
2D-3D co-seg. [18]	35	50	58	71
Our method	104	74	63	64

Table 2: The number of times a specific method was chosen to best convey the same meaning as the original drawing in the qualitative experiment using Amazon Mechanical Turk. Both outside-in (OutIn_1 and OutIn_2) and inside-out (InOut_1 and InOut_2) scenes were used. The proposed method had more votes than the other two methods except in the InOut_2 dataset.

In addition to the user study, we note that Nuernberger *et al*. [18] also evaluated annotation transfer results by the Intersection-over-Union (IoU) score – in a transfer view, calculating how well the rendered circle overlaps the selected object rather than irrelevant background. This may not be a good metric in a more complicated test scene like what we present in this paper. Figure 4 is an example: a tower occludes the middle of the target Arc de Triomphe in the transfer view so a well rendered circle that can cover both the left and the right parts of the Arc de Triomphe must inevitably overlay the tower, thus resulting in a low IoU score.

7 Discussion and Future Work

The proposed method generally produced satisfactory segmentation and rendering circles of higher subjective quality. However, as with most interactive segmentation methods, we observed that many times both our method and Nuernberger *et al*.'s 2D-3D co-segmentation method [18] struggle with filtering out the background scene whenever the user draws a contour that includes too much background content (see Figure 8 for an example). In such cases, the 2D ellipse, fitted to the projected 3D segmented points, may become skewed due to the false positive 3D segmented points. This inspired us to consider an alternative visualization method to the 2D ellipse—a simple highlight of the object, using the segmented 3D points. With such a visualization, the user may more easily understand that several false positive segmented points are actually outliers and are thus meaningless to the object reference. Future work should explore what kind of visualizations are most robust in these cases and in cases where the object of interest has a complex shape. Furthermore, how to handle under- and over-segmentation appropriately in an interactive way should be explored.

While this paper demonstrates an initial success of our approach with Elastic Fusion [27], which yields a dense surfel-based representation of the scene, future work may apply the proposed method on semi-dense or sparse models. It could also be valuable to create a dataset with many user drawn circles as well as manually labeled 3D ground-truth segmentations for a variety of models reconstructed in different fashions. With such a dataset, a better set of parameters of our algorithm could be automatically learned and

cross-validated, and the sensitivity of the parameters to the model density or other model properties could be quantitatively studied.

While our experiments in this paper focused on comparing to previous methods [4, 18] in an object referencing scenario, we see our method as being applicable to a wider set of AR applications rather than only being used for transferring gesture annotations between viewpoints. Segmentation via 2D input may be used for localizing objects of interest for further interaction, such as for games, labeling of objects [28], and more. Future work should explore how segmentation methods such as ours affect the AR collaboration performance in a variety of scenarios, including remote, asynchronous, and co-located collaboration.

8 Summary

We presented a novel three-layer graph-based segmentation method for object referencing in collaborative augmented reality. First, we discussed two major challenges of appropriately handling 2D on-screen circle referencing gestures for 3D object referencing in AR. Next, we introduced our novel unified framework that handles both outside-in and inside-out scenes, both convex and planar objects. Specifically, we minimize an energy function using graph-cuts on a three-layer graph composed of pixel, 3D point, and supervoxel energies. Experimental results with our prototype showed that our method outperformed previous approaches in three out of four datasets we captured, including both inside-out and outside-in scenes.

Acknowledgements

We thank Domagoj Baričević for discussions and experiments on Kinect sensors. We thank Donghao Ren for helping preparing experimental datasets. This work was partially supported by Citrix Online, ONR grant N00014-14-1-0133, and the National Science Foundation under Grant No. 1219261.

References

[1] A. Blake, P. Kohli, and C. Rother. *Markov random fields for vision and image processing*. MIT Press, 2011.

[2] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(9):1124–1137, 2004.

[3] S. Gauglitz, C. Lee, M. Turk, and T. Höllerer. Integrating the physical environment into mobile remote collaboration. In *Proceedings of the 14th international conference on Human-computer interaction with mobile devices and services*, pages 241–250. ACM, 2012.

[4] S. Gauglitz, B. Nuernberger, M. Turk, and T. Höllerer. In Touch with the Remote World: Remote Collaboration with Augmented Reality Drawings and Virtual Navigation. In *Proceedings of the 20th ACM Symposium on Virtual Reality Software and Technology*, pages 197–205, New York, NY, USA, 2014. ACM.

[5] A. Golovinskiy and T. Funkhouser. Min-cut based segmentation of point clouds. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 39–46. IEEE, 2009.

[6] P. Gurevich, J. Lanir, B. Cohen, and R. Stone. TeleAdvisor: A Versatile Augmented Reality Tool for Remote Assistance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 619–622, New York, NY, USA, 2012. ACM.

[7] D. D. Hoffman and W. A. Richards. Parts of recognition. In *Cognition*, pages 65–96. Elsevier, 1984.

[8] C. Hoppe, M. Klopschitz, M. Donoser, and H. Bischof. Incremental Surface Extraction from Sparse Structure-from-Motion Point Clouds. In *Proceedings of 24th British Machine Vision Conference*, BMVC '13, pages 94.1–94.11, Bristol, UK, 2013. BMVA Press.

[9] O. V. Kaick, N. Fish, Y. Kleiman, S. Asafi, and D. Cohen-Or. Shape segmentation by approximate convexity analysis. *ACM Transactions on Graphics (TOG)*, 34(1):4, 2014.

[10] S. Kasahara, V. Heun, A. S. Lee, and H. Ishii. Second Surface: Multi-user Spatial Collaboration System Based on Augmented Reality. In *SIGGRAPH Asia 2012 Emerging Technologies*, SA '12, pages 20:1–20:4, New York, NY, USA, 2012. ACM.

[11] S. Kim, G. A. Lee, N. Sakata, A. Dunser, E. Vartiainen, and M. Billinghurst. Study of augmented gesture communication cues and view sharing in remote collaboration. In *Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on*, pages 261–262. IEEE, 2013.

[12] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, Nara, Japan, November 2007.

[13] J. Liang and M. Green. Jdcad: A highly interactive 3d modeling system. *Computers & Graphics*, 18(4):499 – 506, 1994.

[14] K.-C. Lien, B. Nuernberger, T. Höllerer, and M. Turk. [poster] 2d-3d co-segmentation for ar-based remote collaboration. In *Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on*, 2015.

[15] G. P. Meyer and M. N. Do. 3d grabcut: interactive foreground extraction for reconstructed 3d scenes. In *Proceedings of the 2015 Eurographics Workshop on 3D Object Retrieval*, pages 1–6. Eurographics Association, 2015.

[16] O. Miksik, V. Vineet, M. Lidegaard, R. Prasaath, M. Nießner, S. Golodetz, S. L. Hicks, P. Pérez, S. Izadi, and P. H. Torr. The semantic paintbrush: Interactive 3d mapping and recognition in large outdoor spaces. In *CHI*, pages 3317–3326. ACM, 2015.

[17] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, pages 127–136. IEEE, 2011.

[18] B. Nuernberger, K.-C. Lien, T. Höllerer, and M. Turk. Interpreting 2d gesture annotations in 3d augmented reality. In *IEEE 3DUI*, 2016.

[19] O. Oda and S. Feiner. 3D referencing techniques for physical objects in shared augmented reality. In *ISMAR*, pages 207–215. IEEE, nov 2012.

[20] J. Papon, A. Abramov, M. Schoeler, and F. Worgotter. Voxel cloud connectivity segmentation-supervoxels for point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2027–2034, 2013.

[21] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. *ACM TOG*, 23, 2004.

[22] R. Salas-Moreno, R. Newcombe, H. Strasdat, P. Kelly, and A. Davison. Slam++: Simultaneous localisation and mapping at the level of objects. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1352–1359, 2013.

[23] M. Tait and M. Billinghurst. The Effect of View Independence in a Collaborative AR System. *Computer Supported Cooperative Work (CSCW)*, 24(6):563–589, 2015.

[24] M. Tatzgern, R. Grasset, D. Kalkofen, and D. Schmalstieg. Transitional augmented reality navigation for live captured scenes. In *Virtual Reality (VR), 2014 IEEE*, pages 21–26. IEEE, 2014.

[25] M. Tsang, G. W. Fitzmaurice, G. Kurtenbach, A. Khan, and B. Buxton. Boom Chameleon: Simultaneous Capture of 3D Viewpoint, Voice and Gesture Annotations on a Spatially-aware Display. In *Proceedings of the 15th ACM Symposium on User Interface Software and Technology*, UIST '02, pages 111–120, New York, NY, USA, 2002. ACM.

[26] J. Valentin, V. Vineet, M.-M. Cheng, D. Kim, J. Shotton, P. Kohli, M. Nießner, A. Criminisi, S. Izadi, and P. Torr. Semanticpaint: Interactive 3d labeling and learning at your fingertips. *ACM Transactions on Graphics (TOG)*, 34(5):154, 2015.

[27] T. Whelan, S. Leutenegger, R. F. Salas-Moreno, B. Glocker, and A. J. Davison. Elasticfusion: Dense slam without a pose graph. *Proceedings of Robotics: Science and Systems (RSS)*, 2015.

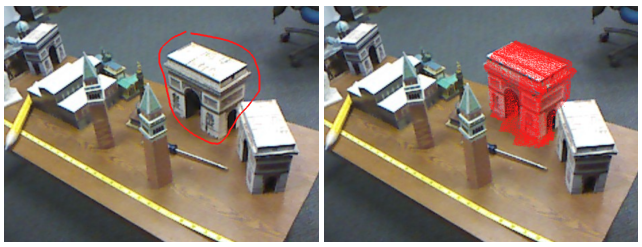
[28] J. Wither, S. DiVerdi, and T. Höllerer. Annotation in outdoor augmented reality. *Computers & Graphics*, 33(6):679–689, 2009.

(a) Original drawing. (b) Median depth plane [4]. (c) 2D-3D co-segmentation [18]. (d) Proposed method.

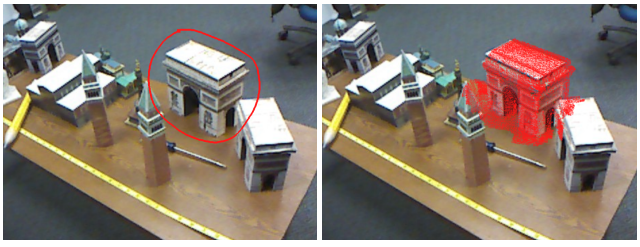
Figure 6: An inside-out experiment result showing the annotation from a drastically different viewpoint. In general, 3D planar interpretations of 2D on-screen gestures can work for mostly planar objects; in this case, however, the orientation of the 3D plane does not help with the annotation rendering from a novel viewpoint. Convex 2D-3D co-segmentation also fails [18]. Only our proposed method gives decent results.

(a) Original drawing. (b) Median depth plane [4]. (c) 2D-3D co-segmentation [18]. (d) Proposed method.

Figure 7: An inside-out experiment result showing the annotation from a drastically different viewpoint. In this case, convex 2D-3D co-segmentation [18] oversegments the region, whereas our proposed method correctly limits the segmentation to the desired area. The AMT votes for this set of images were: 1 for median depth plane, 3 for 2D-3D co-segmentation, and 16 for the proposed method.



(a) Original drawing. (b) Background is correctly discarded.



(c) Original drawing. (d) Foreground is incorrectly kept.

Figure 8: Top row: example where our method can correctly discard the background in the distance. Bottom row: example where our method incorrectly keeps foreground points in the segmentation.