Interactive Segmentation on RGBD Images via Cue Selection

Jie Feng[1] Brian Price[2] Scott Cohen[2] Shih-Fu Chang[1]

[1]Columbia University
{jiefeng, sfchang}@cs.columbia.edu

[2]Adobe Research
{bprice, scohen}@adobe.com

Abstract

Interactive image segmentation is an important problem in computer vision with many applications including image editing, object recognition and image retrieval. Most existing interactive segmentation methods only operate on color images. Until recently, very few works have been proposed to leverage depth information from low-cost sensors to improve interactive segmentation. While these methods achieve better results than color-based methods, they are still limited in either using depth as an additional color channel or simply combining depth with color in a linear way. We propose a novel interactive segmentation algorithm which can incorporate multiple feature cues like color, depth, and normals in an unified graph cut framework to leverage these cues more effectively. A key contribution of our method is that it automatically selects a single cue to be used at each pixel, based on the intuition that only one cue is necessary to determine the segmentation label locally. This is achieved by optimizing over both segmentation labels and cue labels, using terms designed to decide where both the segmentation and label cues should change. Our algorithm thus produces not only the segmentation mask but also a cue label map that indicates where each cue contributes to the final result. Extensive experiments on five large scale RGBD datasets show that our proposed algorithm performs significantly better than both other color-based and RGBD based algorithms in reducing the amount of user inputs as well as increasing segmentation accuracy.

1. Introduction

Binary image segmentation is the process of separating pixels into foreground and background. It is an important problem for many computer vision applications, e.g. image editing, object recognition, image retrieval, etc. Automatic segmentation is intrinsically ambiguous and thus cannot obtain satisfactory results on an arbitrary image without any high-level understanding of the content. On the other hand, interactive image segmentation allows a user to tell the algorithm what should be selected or not. An ideal interactive segmentation algorithm should: 1) require minimal amount of user interaction; 2) achieve good accuracy. However, the colors in an image are affected by illumination, appearance, occlusion, etc., making them less reliable for the segmentation task. Due to this fact, significant effort from users is still necessary to achieve satisfying results on complex images.

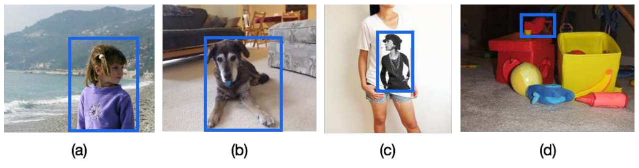


Figure 1: Example foreground/background cases. (a) complex appearance, clean depth separation; (b) touching surface; (c) same surface, different appearance; (d) touching surface, similar appearance, background clutter.

Recent years have witnessed the emergence of low-cost depth sensors, such as Microsoft Kinect, Intel Realsense and Google Project Tango. These sensors are able to acquire a depth image which captures the physical distance of the scene to the camera at each pixel. This information is very useful for image segmentation but is lost in color imaging process. Besides depth, other feature cues can also be extracted from a depth image to describe the scene, e.g. normal map, 3D point cloud, mesh structure, etc. Together with paired RGB color image, an RGBD image allows the possibility of combining multiple complimentary cues for the interactive segmentation problem to reduce user input while maintaining or even improving accuracy.

Despite this, few works [5] [6] have been published on interactive segmentation on RGBD images. These works either treat depth as an additional color channel or simply perform a global linear combination of different cue confidences to produce the final result, allowing them to achieve better performance than if using color alone. However, objects and scenes in the real world are complicated, as shown in Fig. 1. Mixing depth and color cues in a fixed way can

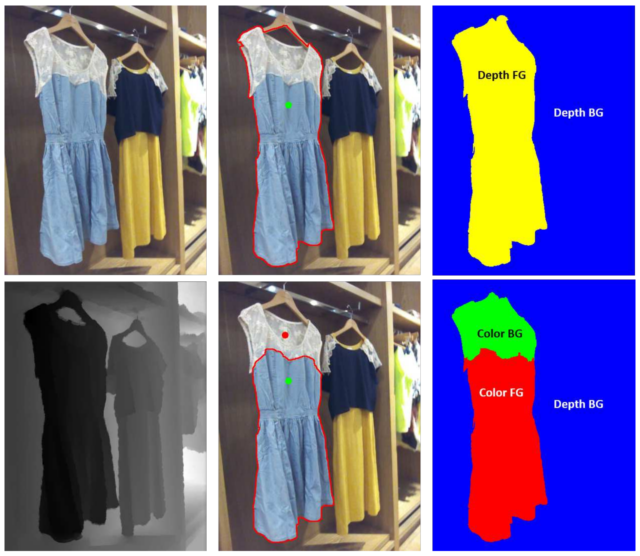


Figure 2: Example result of our algorithm. Foreground click is colored in green and background click is red.

reduce the original discriminative power of each individual cue and not allow the algorithm to adapt, e.g. when an object and background are separated in depth (Fig. 1a), the depth is more useful in determining the foreground than when selecting a color region on a single surface (Fig. 1c). Such methods also provide no way of knowing which cues where most useful for achieving a given segmentation.

To better handle and integrate multiple cues, we propose a novel cue-selection-based interactive RGBD segmentation algorithm within a graph cut framework. We observe that at least one cue will have highest confidence in distinguishing foreground and background locally, which means only one cue per pixel is necessary to infer the final segmentation result. Thus, we convert the standard binary labeling problem into a multi-label problem with each label representing both the segmentation label (foreground or background) and cue label (color, depth, normal). This allows different cues to take effect in different areas of the image, and allows the algorithm to respond to the individual image and the user input. An example result of our algorithm is given in Fig. 2 where in the first row the algorithm segments out the whole dress based on the depth cue given the first foreground click since the depth cue gives a clear separation of the dress and background. By adding a background click on the upper white part, our algorithm can intelligently obtain only the lower blue part by switching to use the color cue as shown in the second row.

Our primary contribution is this multi-cue-selection-based interactive selection paradigm as applied to RGBD image selection. We model the foreground/background probablities using a geodesic-distance-based adaptive foreground confidence map. The pairwise term is designed to ensure smoothness of both segmentation label and cue label. Alpha-beta swap is used to efficiently find the optimal labels. Our approach is similar to applying a dynamic binary weighting among multiple cues at each pixel location where only one cue gets a non-zero cue weight. To evaluate our algorithm, extensive experiments are conducted on five large scale RGBD datasets captured from different depth sensors. Our method is able to achieve similar or better segmentation accuracy with significant fewer user inputs when comparing with color based algorithms and other RGBD based algorithms.

2. Related Works

2.1. Interactive segmentation on color images

A large body of work has been proposed for segmentation on color images. Among them, the graph cut based framework has been very popular since it was introduced by Boycov et. al [18]. In their work, the image is represented by a graph and user inputs act as hard constraints. Graph cut is used to find a globally optimal segmentation based on an energy function with balanced region and boundary information. However, [18] is limited in its reliance on color information only, and so can fail in cases where the foreground and background color distributions are overlapping or complicated. Various works tried to improve on this color-based segmentation. GrabCut [14] iteratively updates a Gaussian Mixture Model (GMM) of the foreground and background to try to improve the segmentation. Bai et al. [2] uses geodesic paths instead of graph cut to avoid its boundary-length bias. Price et al. [13] combines geodesics with graph cut to try to take advantage of their relative strengths. Gulshan et al. [7] introduces a star-convexity shape constraint to work with geodesic distances for segmentation. While these enhancements can improve the interactive experience, they are still limited to the fact that they rely on the color information only and so still struggle in cases of overlapping or complex foreground/background color distributions.

2.2. Interactive segmentation on RGBD images

Unlike the flourish of interactive segmentation methods for color images, there are very few works about interactive segmentation on RGBD images. Diebold et al. [5] utilizes a segmentation formulation based on total variation. Depth is added as an additional color channel and a joint distribution for foreground pixels is computed by incorporating three Gaussian kernels for distance, color and depth respectively. This method extends the spatially varying color distributions [11] using 3D geometry and the distance is also computed using depth information. Experiments on a small RGBD dataset shows the proposed method achieves better segmentation quality with less user scribbles required. In another recent work, Ge et al. [6] employs a binary graph cut framework where color and depth cues are used separately to compute costs that are linearly combined as the final unary term. Histogram and geodesic distance are used for each cue respectively. A hierarchical image pyramid is also used to speed up graph cut process. Experiments performed on the RGBD saliency dataset [12] and a stereo dataset demonstrates better results than [5].

Although the above methods show the obvious advantage of using RGBD image for interactive segmentation, they either add depth as an additional color channel to compute the foreground confidence or take a simple linear combination with color and depth cues. This additive nature can be problematic when only one cue is useful in segmentation, e.g. different colors on the same depth surface or similar color for foregrounds and backgrounds or differing depth. Adding the cues directly can reduce the discriminability of cues overall thus making it harder to produce good results with limited user inputs. Our approach instead selects only one cue to determine the segmentation locally, resulting in a practical and general method for fusing multiple cues while preserving the original foreground confidence for each cue.

3. Approach

We first introduce the basic graph cut framework for color image segmentation. Then we describe how this framework can be adopted for our RGBD segmentation with cue selection capability.

3.1. Binary MRF for Interactive Segmentation

Let i denote a pixel in image I and Ω denote the set of all pixels in I. N is the set of adjacent pixel pairs. Interactive image segmentation is the problem of dividing Ω into two disjoint sets, Ω_1 for foreground and Ω_0 for background, given some user inputs. It is usually formulated as a binary labeling problem via Markov Random Field (MRF) with the following energy function:

$$E(S) = \sum_{i \in \Omega} D(S_i) + \lambda \sum_{(i,j) \in N} f(S_i, S_j) \tag{1}$$

where S_i is the segmentation label for pixel i and S is the labeling of all pixels. S_i takes the value of 0 or 1 to indicate the pixel belongs to background or foreground respectively. In Eq. 1, $D(S_i)$ represents the cost to assign label S_i to pixel i. It is often referred to as unary term which usually takes the form of:

$$D(S_i) = -\log P(S_i) \tag{2}$$

where $P(S_i)$ is the probability of pixel i being assigned to label S_i. For user specified foreground or background pixels, the probability is set to 1 for corresponding labels to make sure the final result obeys user inputs. $f(S_i, S_j)$ is the cost for assigning a pixel pair S_i and S_j and is referred to as the pairwise term. In the color image case, $f(S_i, S_j)$ has the form:

$$f(S_i, S_j) = \begin{cases} 0 & \text{if } S_i = S_j \\ g(i,j) & \text{if } S_i \neq S_j \end{cases} \tag{3}$$

with the similarity between adjacent pixels given by $g(i,j) = \exp(\frac{-|I_i - I_j|^2}{2\sigma^2})$ where I_i is the color pixel value for pixel i, so the cost will be small if we assign different labels to nearby pixels with low similarity. λ controls the balance between unary and pairwise terms. By minimizing Eq. 1, we are able to get the optimal pixel labeling S^*. Graph cut [3] can be used to efficiently minimize this energy function.

3.2. RGBD Segmentation as Multi-label MRF

To better handle complex natural scenes, we use three different cues to infer foreground confidence for each pixel. Color is useful for identifying foregrounds with different appearance from their backgrounds. Given a depth map, we extract two types of cues for each pixel. First, we use the depth value directly as one cue since it gives important information about the relative spatial distance between foreground and background. Furthermore, to be able to distinguish objects with different geometry but similar distance, normal vectors are computed from a depth-projected 3D point cloud.

To tackle segmentation by fusing multiple cues, we propose a cue selection approach based on the assumption that given certain user inputs, only one cue is required to explain the segmentation result for each pixel. More specifically, we want to know 1) if the pixel is foreground or background; 2) which cue is most discriminative in determining the labeling. This assumption aligns well with our intuition, allows our algorithm to determine how to apply the cues on a local basis, and also provides an interpretation of how each cue contributes to the segmentation. Based on this motivation, we form a label pair $X_i = < S_i, C_i >$ for each pixel i. S_i is the segment label which takes the value of 0 or 1 to indicate if the pixel is background or foreground respectively. C_i is the cue label which takes the value from 0 to N-1 if there are N cues. By linearizing the label pair into a label within a $[0, 2 \times N)$ range, we can reformulate the labeling problem as a multi-label MRF.

$$E(X) = \sum_{i \in \Omega} D(X_i) + \lambda \sum_{(i,j) \in N} f(X_i, X_j) \tag{4}$$

To adapt this model to our problem, we need to provide appropriate energy terms.

3.2.1 Foreground/Background Confidence Maps

For the unary term, it is similar to the binary case where each label will have a corresponding cost:

$$D(X_i) = 1 - P_{C_i}(S_i) \tag{5}$$

The probability $P_{C_i}(S_i)$ is based on a confidence map for cue C_i that indicates how likely a pixel belongs to foreground or background for that individual feature. We found this form works better than the log form as in Eq. 2. We compute this confidence map using a geodesic distance transform.

The geodesic distance transform has been used in interactive segmentation methods [2] [13] [7] to achieve very promising results. Geodesic distance inherently encodes spatial information between pixels and is good at separating regions with similar feature values that are not adjacent. This property fits well with interactive segmentation where the target is usually a single connected component. Also, since a depth map shows the physical connectivity of pixels, using geodesic distance can produce a more useful distance measure between pixels and user inputs.

We compute foreground and background confidence map given user inputs U where U_1 denotes foreground pixels and U_0 the background pixels as indicated by the user. We first construct a weighted graph $G = (V, E)$, where V is a set of nodes and E the edges between nodes. The weight is computed using a different distance measure for each feature cue. For color, we convert RGB value into LAB space and use L2 norm as distance. For depth, the absolute difference between depth values is used. As for normal, cosine similarity is used to compute the similarity score between two unit normal vectors and then converted to a distance measure. The geodesic distance between any two pixels i and j is essentially the length of the shortest path $d(i, j)$ on G. It can be computed exactly with Dijkstra's algorithm. For each pixel i, we calculate a geodesic distance to the closest foreground pixel as $d(i, U_1) = \min_{j \in U_1} d(i, j)$. Similarly for background, we get $d(i, U_0)$. Then we convert the two values into a probability measure:

$$P_{C_i}(S_i) = \frac{d(i, U_{\bar{S}_i})}{d(i, U_{S_i}) + d(i, U_{\bar{S}_i})} \tag{6}$$

where $\bar{S}_i$ is the opposite label of S_i (e.g. if $S_i = 0$ then $\bar{S}_i = 1$). This is done for each feature cue independently.

3.2.2 Multi-cue Pairwise Term

The pairwise term is similar to the single-cue case in Eq. 3 except modified to handle multiple cues:

$$f(X_i, X_j) = \begin{cases} 0 & \text{if } C_i = C_j \text{ and } S_i = S_j \\ g'(i, j) & \text{otherwise} \end{cases} \tag{7}$$

where

$$g'(i, j) = \min\left(\exp\left(\frac{-D^2_{C_i}(i, j)}{2\sigma^2_{C_i}}\right), \exp\left(\frac{-D^2_{C_j}(i, j)}{2\sigma^2_{C_j}}\right)\right) \tag{8}$$

and D_{C_i} is the raw feature distance between adjacent nodes for cue C_i as computed in geodesic distance transform.

In the case that $C_i = C_j$, this is the same as Eq. 3 being applied to the feature C_i. No cost is assigned if the segment label does not change, and a cost inverse to the edge strength is assigned if different segment labels are given. In the case that $C_i \neq C_j$, we want the algorithm to always choose the most discriminative cue to decide where to place the boundary versus where to enforce smoothness in the labels. Thus the cost is based on the inverse edge strength of the best cue possible.

3.2.3 Optimization

Given the unary and pairwise terms, we combine them into the MRF formulation in Eq. 4. Directly minimizing the energy function is an NP-complete problem in general. There exist methods to search for a local minimum such as alpha expansion [4] and alpha-beta swap [4]. Since our pairwise term is not a sub-modular function, it can not be optimized by alpha expansion. Thus alpha-beta swap is adopted. This algorithm randomly selects two labels from the label set and tries to reduce the energy by swapping these labels. It usually only takes a few iterations (3-4) to converge.

To ensure good responsiveness for interactive segmentation, we use superpixels instead of pixels as nodes in our graph to reduce the node numbers. We use SLIC [1] to compute our superpixels due to its regular shaped results and good efficiency. The feature vector of each cue on a superpixel is the mean value of all pixel features. Superpixels are extracted from color image given that depth map is well aligned. The mean feature vector for a superpixel is robust to light misalignment between color and depth. The pixel level confidence map can be created by setting all pixels within the same superpixel to the superpixel score.

Superpixels may not exactly follow the object boundary at the pixel level. To produce an accurate and smooth boundary, a pixel-level boundary refinement is performed after obtaining the segmentation mask. Only superpixels along the foreground and background boundary are allowed to change. We set pixels within these superpixels to have unknown labels. All other pixels are set to either hard foreground or background based on their corresponding superpixels. GrabCut [14] is used to infer the labels for boundary pixels.

3.2.4 User Interaction

In an interactive segmentation problem, user inputs typically can take one of the following forms: 1) foreground/background clicks: this gives the least amount of user inputs; 2) foreground/background strokes: this is essentially a series of clicks; 3) bounding box around target object: strong indication of background outside the box and weak indication of object inside the box. These three forms of user inputs can all be converted to a pixel-level input to be used in our problem setting. In this work, we look at the first input type, fg/bg clicks, as it brings the best user experience by imposing the least amount of effort at user side and at the same time is the most challenging one with the least supervision for a segmentation algorithm.

When starting, a user may provide a foreground click before any background clicks. When this happens, the foreground probability will be computed as $d(i, U_1)$ normalized by $\max_i d(i, U_1)$. Because of the cumulative nature of geodesic distance, even the pixels within the same object may receive a big distance value if they are far away from the foreground pixels. To deal with this issue, we adopt a background prior similar with that used in salient object detection [17]. We treat superpixels touching the image boundary as potential background. To deal with cases where the object is touching the image boundary, we compute a likelihood score for each superpixel. Unlike the 1D saliency measure used in [17], the additional depth map provides much stronger evidence of how a boundary superpixel is connected to a foreground superpixel. We use a predefined threshold (0.5) on depth geodesic distance between boundary superpixels and the closest foreground superpixel. Those having a larger distance are classified as background.

4. Experiments

4.1. Datasets

We evaluate our RGBD interactive segmentation algorithm on five datasets captured by two popular depth sensors, Microsoft Kinect v1 and Kinect v2. Kinect v1 uses structured light for measuring depth while Kinect v2 uses Time-of-Flight (ToF) to get higher fidelity depth values. For Kinect v1 datasets, we used 3 public datasets: RGBD Salient Object dataset (Saliency) [12], Berkeley 3D dataset (B3D) [8] and NYU Depth2 dataset (NYU2) [15]. The RGBD salient object dataset contains 1000 images with corresponding depth images. Both indoor and outdoor scenes are covered. Each image contains only one annotated object. The Berkeley dataset contains 849 RGBD images from indoor scenes, each image usually containing more than one object. It was originally collected for object detection, and we use a subset of 554 images with the object masks from [16]. The NYU2 dataset contains 1449 RGBD images mostly of cluttered scenes that have many small objects. Since the dataset provides category labels for each pixel, we extract object masks from these annotations. For Kinect v2 datasets, we use those provided by the SUN RGBD benchmark suite [16]. The benchmark includes two Kinect v2 datasets, one with 300 images (alignedkv2) and the other with 3486 images (kv2data).

We use all annotated objects in each dataset as a target to segment. To the best of our knowledge, this is by far the largest evaluation task for interactive segmentation both on color and RGBD images. These datasets exhibit diverse real world scenes with various object categories, making segmentation very challenging.

4.2. Evaluation Protocol

For interactive segmentation, the most important performance metrics are segmentation accuracy and the amount of user inputs required. We use the following two protocols to cover both cases.

4.2.1 Accuracy given fixed inputs

To compute accuracy of an output segment given a ground truth segment, we use Jaccard coefficients as in prior works [9] [6]. It computes the ratio of areas from segment intersection and segment union, aka IoU (Intersection over Union). We give the same inputs to all methods for fair comparison. The inputs are computed as skeleton pixels of both foreground and background regions for each ground truth mask. An example input is shown in Fig. 3.

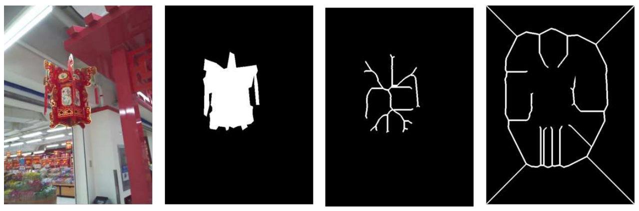


Figure 3: Example skeleton input. From left to right: color image, ground truth mask, foreground skeleton, background skeleton.

4.2.2 Accuracy vs. Clicks (AvC)

To show the effort required by a user for the interactive segmentation task, we calculate the average IoU value given a certain number of clicks. In order to conduct large scale evaluation in an automatic way, we designed a method to predict the next best click adaptively given the output segment mask and ground truth mask. First, we compute the intersection of the ground truth and current segmentation mask. We subtract it from the ground truth to obtain regions that need to be added, and subtract it from the current selection to obtain regions that need to be removed. Considering disjoint regions may exist in the mask, we extract the largest connected component from it and compute its centroid. The valid pixel closest to the centroid is selected as the next click with corresponding fg/bg label. This algorithm is guaranteed to eventually converge to the ground truth. Given an input image, we iteratively predict the next best pixel to click and get an output segment mask. IoU is then computed. The process terminates when the maximum click number is reached and we keep tracking of the average IoU given the number of clicks on all images from each dataset. This metric is referred to as AvC.

4.3. Comparison Methods

We compare with segmentation algorithms on both color and RGBD. For color, we compare with 3 algorithms: 1) GrabCut [14]; 2) Lazy Snapping [10]; 3) Geodesic Graph Cut [13]. For RGBD based segmentation, we look at two interactive segmentation methods which were published very recently, RGBD Linear Comb [6] and RGBD TVSeg [5]. We use the GrabCut implementation from OpenCV by changing the bounding box input to a pixel mask input. Three iterations are applied for each cut. The Geodesic graph cut code is kindly provided by the authors. RGBD TVSeg has an open source implementation. For all other methods, we implemented them ourselves.

For our method, we show results using only color and depth cues, and using color, depth, and normal cues. This allows more fair comparison to the RGBD methods that do not use normals.

4.4. Implementation Details

Normals are computed using the average cross-product of vertex vectors within a neighborhood. It gives similar accuracy to plane fitting but runs more efficiently. For each image, we generate approximately 800 superpixels to have accurate separation along the object boundary. The multi-label MRF optimization is carried out using the gco library from [3]. The pairwise weight is set to 0.1 which we find produces the best results for our algorithm.

4.5. Results

We first show segmentation accuracy given fixed input on three Kinect v1 datasets in Table 1. Our method achieves consistently higher accuracy compared to both color and RGBD methods. Close to 10% improvement can be seen by effectively incorporating depth compared to color only methods. Our method achieves an average $3-5\%$ higher accuracy compared to other RGBD based algorithms.

The first row in Fig. 4 shows the AvC curves on Kinect v1 datasets. Our algorithm is able to get best performance across all datasets with a clear margin. On RGBD saliency dataset, due to its relative simplicity, all methods perform much better than on B3D and NYU2. To achieve an IoU accuracy of 90%, our method only needs 5 clicks compared to 15 clicks required by the best color model Geodesic Graph Cut. With the same amount of clicks, the proposed method gets about 8-10% higher for absolute accuracy than competing RGBD segmentation methods. The other two datasets are much more challenging due to small objects, clutter and noisy depth input, so the performance of all methods decrease noticeably. Yet, our method is able to improve its accuracy more stably while clicks are added. Our method gets much better accuracy when only a few clicks (5-10) are present. By adding normal cue, the performance of our algorithm can be further improved on almost all datasets. Up to 5% gain can be seen on B3D dataset. This shows the power of cue selection which fuses multiple cues for segmentation while preserving the discriminability of each cue. We use RGBD Saliency dataset as a testbed to further conduct comparisons with two additional methods. First is a recent color based method [11]. It performs much better than other color based methods but still fall behind our method with color and depth cues by a margin. The other method is binary graph cut using geodesic distance on the joint feature vector of each cue. It manages to beat other two RGBD approaches but is clearly worse than our method both using color and depth or all three cues. This indicates the proposed multi-label formulation for cue selection is more advanced than binary energy which is also limited to produce only segmentation label.

Method	Saliency	B3D	NYU2
GrabCut [14]	0.78	0.65	0.58
Lazy Snapping [10]	0.75	0.60	0.50
Geodesic Graph Cut [13]	0.80	0.71	0.64
RGBD TVSeg [5]	0.84	0.76	0.73
RGBD Linear Comb [6]	0.85	0.77	0.74
Ours: Color+Depth	0.87	**0.82**	0.77
Ours: Color+Depth+Normal	**0.87**	0.81	**0.78**

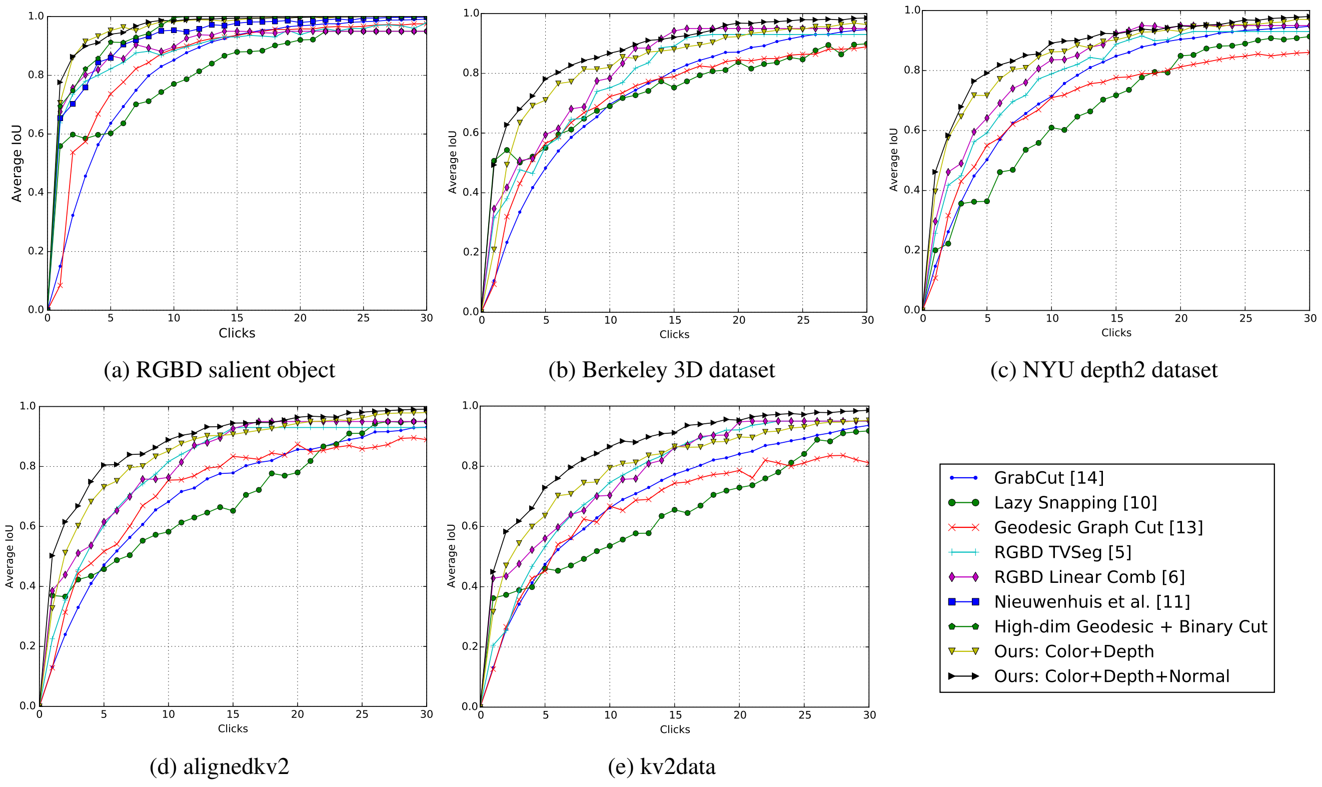
Table 1: IoU values with fixed inputs on Kinect v1 datasets.

Method	Aligned KV2	KV2data
GrabCut [14]	0.56	0.58
Lazy Snapping [10]	0.52	0.51
Geodesic Graph Cut [13]	0.55	0.53
RGBD TVSeg [5]	0.65	0.60
RGBD Linear Comb [6]	0.66	0.59
Ours: Color+Depth	0.72	0.66
Ours: Color+Depth+Normal	**0.73**	**0.68**

Table 2: IoU values with fixed inputs on Kinect v2 datasets.

Besides the three Kinect v1 datasets, we also evaluated all methods on two Kinect v2 datasets. The fixed input accuracy is shown in Table 2. Again, our algorithm has much higher performance comparing to other color based and RGBD based algorithms.

The second row in Fig. 4 plots AvC curves on Kinect



(a) RGBD salient object (b) Berkeley 3D dataset (c) NYU depth2 dataset

(d) alignedkv2 (e) kv2data

Figure 4: AvC on 5 RGBD datasets (best viewed in color).

v2 datasets. On both datasets, our method performs significantly better than all other methods. For alignedkv2, at least 12% improvement can be seen when only 5 clicks are given. With the inclusion of the normal cue, our accuracy under same number of clicks improves about 5%. This is due to the existance of many objects touching the ground like chairs and tables. Noticeably, our method also shows better stability (less zigzag up and down) when more clicks are given as compared with other methods, including the two RGBD based methods.

We show example segmentation results from each dataset in Fig. 5. These examples cover various real world scenarios where selection can be challenging. Row 1 shows a case where the object has similar color as the background. By leveraging depth, it can be selected by our algorithm effortlessly. In row 2, depth information is almost useless in selecting the world map on the wall. After placing a background click outside it on the wall, the algorithm knows the color cue is more useful to distinguish foreground from background and uses it for the selection.

Rows 3-6 contain objects that are within complex backgrounds and are not uniformly colored, especially for the footrest and bed in row 5 and 6 where other objects, e.g. magazine, toys are also part of them. Depth alone is not sufficient due to the similar depth where they touch the ground. However, our multi-cue paradigm is able to combine different cues to get the best selection results with only 1-2 clicks. Note the cue switch in rows 4 and 5 where a transition from depth cue to normal cue happens inside the object. This allows the algorithm to use the more discriminative normal cue for separating the ground and the object.

Rows 7 and 8 showcase selection in a cluttered scenes with large color and depth variation. There are also other objects with similar appearance and depth close to our targets. Our algorithm still performs well, requiring only few clicks. In the last row, the depth image is very noisy and even incorrect in some image regions, e.g. upper left background. Our algorithm is quite robust to avoid incorrect background depth and use the more confident color cue to determine background region while still leveraging the depth cue to obtain the complete multi-colored traffic sign with only 1 click.

This adaptive cue selection characteristic makes our algorithm very effective in producing good segmentation results based on user intention. Additionally, by looking at the cue map, we can understand how each feature cue is behaving to get the final results and which cues are more useful in the given scene context.

Failure Cases While we can always converge to a desired result given enough user interaction, some images re-

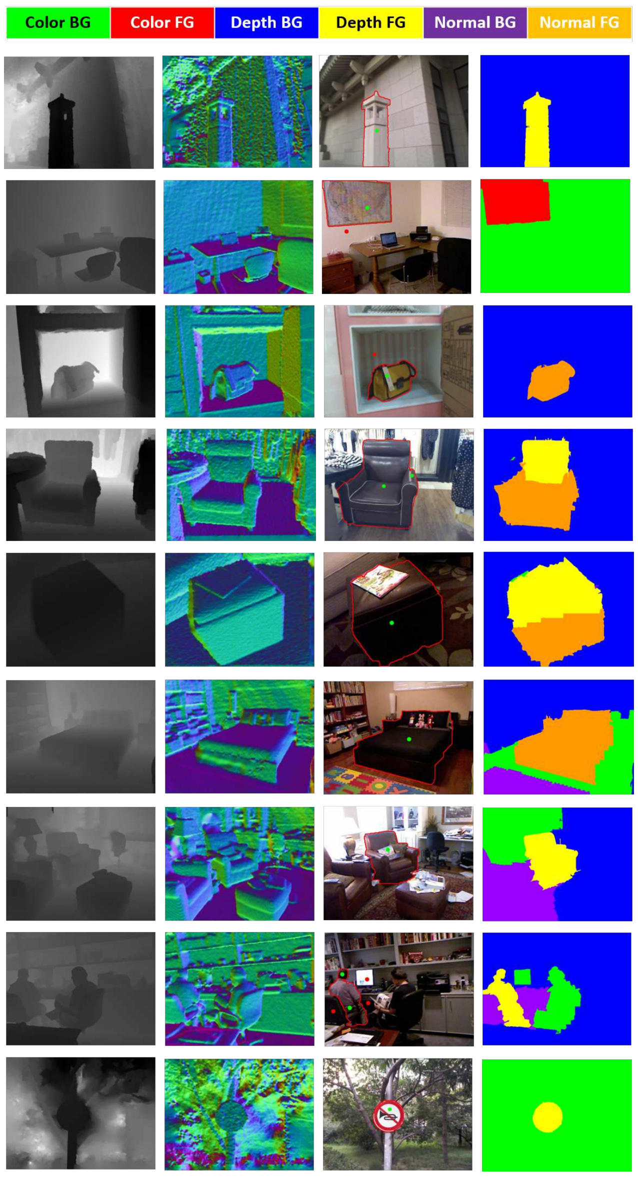


Figure 5: Example results of our RGBD segmentation algorithm. Foreground click is colored green and background click is red (note the input is only the clicked pixel, the click circle is only for visualization). Each row is an example, from left to right: depth image, color coded normal map, input clicks and result contour, feature cue map. The meanings of cue labels are shown in the top bar.

quire more interaction than desired. Example failure cases are shown in Fig. 6. Most of the failures we observed can be attributed to two reasons.

The first is highly unreliable depth quality. Row 1 in Fig. 6 shows an example where the scene is captured outdoor. Current low-cost depth sensors cannot deal well with outdoor scenes under direct sunlight. The depth map is

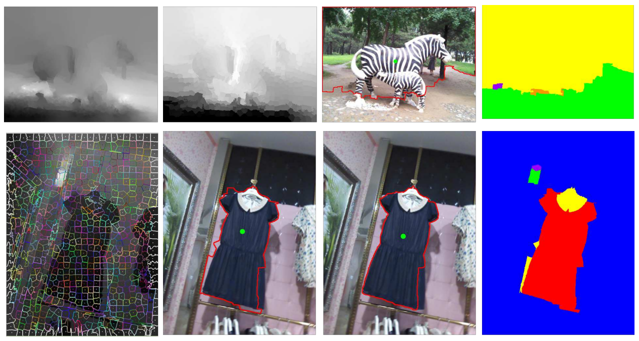


Figure 6: Failure cases. Row 1 (left to right): depth image, foreground confidence on depth map, user click and segmentation result, cue map. Row 2 (left to right): color superpixel contour overlayed on depth image, user click and segmentation result, refined segmentation, cue map.

severely corrupted, which causes our depth foreground map to be incorrect, resulting inaccurate segmentation.

The second reason is misalignment between color and depth. In row 2, with the color superpixel boundary overlaid on top of depth image, it is obvious the superpixels from the color image are not aligned well with the boundary in the depth image, e.g. zoom in to the lower right boundary of the dress. When this happens, our algorithm still tries to use the strongest cue to figure out the boundary, which in this case is depth background. Our boundary refinement is able to recover certain part of the true boundary without adding more inputs.

5. Conclusion

We proposed a novel RGBD interactive image segmentation algorithm based on cue selection. The method effectively fuses multiple feature cues into a unified multi-label MRF framework. Foreground confidence is adaptively computed based on user inputs using a geodesic distance transform. A pairwise term considering both segmentation label and cue label is designed to allow selection of one cue for determining the segmentation result locally while encouraging smooth labeling overall. Extensive experiments on five large scale RGBD datasets captured by Kinect v1 and v2 show our algorithm achieves much better performance than algorithms using only color information by effectively applying depth information. Also, significant improvement is obtained beyond the state-of-the-art RGBD interactive segmentation algorithm.

Future work will be devoted to investigate other important cues for RGBD segmentation and to extend to automatic object segmentation. Additionally, our proposed multi-label MRF is a general approach that can be used given different cues like texture or motion and applied to different problems such as semantic or video segmentation.

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(11):2274–2282, 2012. 4324

[2] X. Bai and G. Sapiro. A geodesic framework for fast interactive image and video segmentation and matting. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007. 4322, 4324

[3] 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. 4323, 4326

[4] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(11):1222–1239, 2001. 4324

[5] J. Diebold, N. Demmel, C. Hazırbaş, M. Moeller, and D. Cremers. Interactive multi-label segmentation of rgb-d images. In *Scale Space and Variational Methods in Computer Vision*, pages 294–306. Springer, 2015. 4321, 4322, 4323, 4326

[6] L. Ge, R. Ju, T. Ren, and G. Wu. Interactive rgb-d image segmentation using hierarchical graph cut and geodesic distance. In *Advances in Multimedia Information Processing–PCM 2015*, pages 114–124. Springer, 2015. 4321, 4323, 4325, 4326

[7] V. Gulshan, C. Rother, A. Criminisi, A. Blake, and A. Zisserman. Geodesic star convexity for interactive image segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3129–3136. IEEE, 2010. 4322, 4324

[8] A. Janoch, S. Karayev, Y. Jia, J. T. Barron, M. Fritz, K. Saenko, and T. Darrell. A category-level 3d object dataset: Putting the kinect to work. In *Consumer Depth Cameras for Computer Vision*, pages 141–165. Springer, 2013. 4325

[9] P. Krähenbühl and V. Koltun. Geodesic object proposals. In *Computer Vision–ECCV 2014*, pages 725–739. Springer, 2014. 4325

[10] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum. Lazy snapping. In *ACM Transactions on Graphics (ToG)*, volume 23, pages 303–308. ACM, 2004. 4326

[11] C. Nieuwenhuis and D. Cremers. Spatially varying color distributions for interactive multilabel segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(5):1234–1247, 2013. 4322, 4326

[12] H. Peng, B. Li, W. Xiong, W. Hu, and R. Ji. Rgbd salient object detection: a benchmark and algorithms. In *European Conference on Computer Vision (ECCV)*, pages 92–109, 2014. 4323, 4325

[13] B. Price, B. Morse, and S. Cohen. Geodesic graph cut for interactive image segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3161–3168. IEEE, 2010. 4322, 4324, 4326

[14] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics (TOG)*, 23(3):309–314, 2004. 4322, 4324, 4326

[15] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgbd images. In *Computer Vision–ECCV 2012*, pages 746–760. Springer, 2012. 4325

[16] S. Song, S. P. Lichtenberg, and J. Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*. 4325

[17] Y. Wei, F. Wen, W. Zhu, and J. Sun. Geodesic saliency using background priors. In *Computer Vision–ECCV 2012*, pages 29–42. Springer, 2012. 4325

[18] B. Yuri and J. Marie-Pierre. Interactive graph cuts for optimal boundaryand region segmentation of objects in nd images. In *IEEEInternational Conference on Computer Vision. USA: IEEE*, volume 112, 2001. 4322