Unsupervised Object Exploration Using Context

Alessandro Pieropan Hedvig Kjellström

***Abstract*— In order for robots to function in unstructured environments in interaction with humans, they must be able to reason about the world in a semantic meaningful way. An essential capability is to segment the world into semantic plausible object hypotheses. In this paper we propose a general framework which can be used for reasoning about objects and their functionality in manipulation activities. Our system employs a hierarchical segmentation framework that extracts object hypotheses from RGB-D video. Motivated by cognitive studies on humans, our work leverages on contextual information, e.g., that objects obey the laws of physics, to formulate object hypotheses from regions in a mathematically principled manner.**

I. Introduction

Reasoning about objects is an essential capability for a robot functioning in unstructured environments. However, the concept of object is vague. According to the Oxford Dictionary, an object is *something that can be touched or seen*. This definition tells little of the nature of an object and it may be subjective. Looking at the same exact image, there can be multiple valid human interpretations of what objects there are [1].

For robotic applications, it is useful for an agent to reason about objects in terms of the current activity [2]. In this paper, *objects are considered to be entities in the world, which can be grasped and moved by a human (or a robot), as part of a manipulation activity*. It should be noted that this is a simplification – what is an object or not is an ill-defined problem, since the nature of human perception is not fully understood.

Our goal is to provide a method to segment scenes into regions, estimate their *objectness* and formulate semantically plausible object hypotheses.

The vision community has spent tremendous effort on segmenting objects in images. Unsupervised object segmentation methods, e.g., [3], [4] are successful when image boundaries coincide with real boundaries between objects in the world, e.g., when objects are uniformly colored with a background in contrasting color, and does not succeed in finding objects with strong texture. This issue is often addressed by introducing supervision in the form of knowledge about the appearance of the objects that are segmented, e.g., [5]. However such a method is confined to detecting objects of previously known classes, and requires training appearance models for this known set of classes. Hence, both approaches have limitations; the first is too generic while the second too specific. We propose here to find a compromise between the two, by leveraging on contextual knowledge. This enables an algorithm flexible enough to accommodate unknown object classes, while expressive enough to formulate valid object hypotheses.

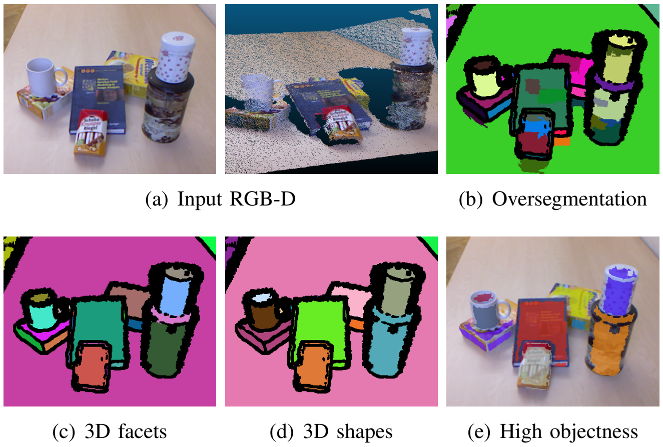


Fig. 1. Unsupervised object discovery. (a) Input RGB-D image. (b) Superpixels generated by an oversegmentation step. (c) Concatenation of super pixels into 3D surface segments, or facets, according to surface orientation. (d) Concatenation of facets into convex 3D shapes. (e) Measuring segment *objectness*; the image shows segments with high objectness.

The main contribution of this paper, illustrated in Fig. 1, is an object discovery algorithm that exploits contextual information to generate 3D segments from an RGB-D image of a scene, whose boundaries correspond to real world boundaries in the scene. Moreover, we propose a mechanism to judge which segments are likely to belong to objects according to the definition above.

II. Related Work

The vision and robotics communities traditionally model objects in terms of object appearance, extracting visual features from images or video and training classifiers to categorize objects in classes [6]. For the purpose of robotic activity modeling, it is beneficial to reason about objects in terms of affordances [2]. This allows the classification of objects in terms of the activites in which they can be employed [7]–[11]. To enable online learning of new object categories (a necessary component in a realistic cognitive robot system), such methods require a method for exploration of scenes and segmentation of object hypotheses, such as the one presented in this paper.

Image segmentation algorithms do not generate object hypotheses per se, but rather divide an image in coherent regions based on color and intensity [3], [12], sometimes also temporal coherence [13] or depth [14], [15].

This research has been supported by the EU through TOMSY, IST-FP7-Collaborative Project-270436, and the Swedish Research Council (VR).

The authors are with CVAP/CAS, KTH, Stockholm, Sweden, pieropan,hedvig@kth.se.

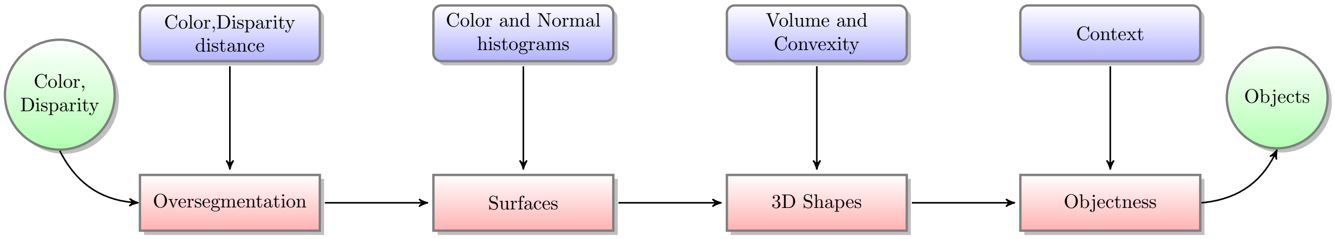


Fig. 2. An RGB-D input image is segmented into superpixels. The resulting oversegmented scene is reprocessed using more descriptive, higher-level features such as color histograms and shape smoothness, resulting in a segmentation into 3D surface segments, or 3D facets. The facets are assembled into convex 3D shapes. These shapes are then analyzed using contextual knowledge to compute the *objectness* of each shape.

Such an approach has a matching between segments and object hypotheses when segment boundaries correspond to real world boundaries. However, when only color information is used such a method cannot disambiguate objects close to each other with similar color appearance or cannot cluster very textured objects.

Recent works have shown that it is possible to have robust matching between segment boundaries and real world boundaries by relying solely on range data. [16] uses only the normals to detect sharp edges and segment the scene using a standard flood fill approach. [17] proposes to use a graph-based segmentation [3] and weight the edges using a local convexity measure based only on disparity data. However both methods may fail when the source of information they rely on is not sufficient to disambiguate complex situations. This can be the case when very thin objects are on top of planar surfaces or objects are aligned presenting no depth discontinuity. [18] proposes to merge segments generated by a color segmentation algorithm [3] with the results of a range based algorithm. However the segmentation on color may lose important edges that correspond to object boundaries, therefore even if the method on disparity generates correct regions there will be no matching with the color segments and vice versa. We believe that range and color data should be used together to enforce the preservation of real world boundaries, crucial for detecting valid object hypotheses. In the spirit of the other methods we use a graph-based segmentation algorithm and we suggest to weight the edges of the graph using all sources of information available. Moreover, as proposed by [16], we leverage on depth discontinuities to modify the connectivity of the graph and enforce the preservation of real world boundaries.

An additional step is required to formulate valid object hypotheses from coherent regions. A way consists of constraining the segmentation and making sure that the segments cohere with appearances of known object categories [5], [19]. The downside is that new categories can not be segmented. Another approach is to provide initial segmentation information in the form of an approximate bounding box around the object region [20] or a point within the region [21]–[24].

Our approach to finding object hypotheses is instead inspired by cognition studies [25]–[27], which stress the importance of context in object detection and recognition. This is spirit of a recent study [28] where volumetric reasoning is applied to an over-segmented scene to improve the pixel-wise segmentation accuracy. We are more interested in formulating valid object hypotheses rather than having an exact segmentation. Therefore we propose a framework, where contextual information is used to judge the *objectness* of the different segments.

III. Contextual Segmentation

As described in the introduction, the present method generates coherent regions using both depth and color data as well as contextual information, suggesting segments with a high likelihood of containing manipulable objects. The method includes four principal steps. In the first step, coherent regions are generated by oversegmenting an RGB-D image using the color and depth measurements as well as surface curvature derived from the depth. In the second, these regions are merged according to surface orientation and color into 3D faces. The third step consists of a procedure where 3D face segments are merged into convex 3D shapes. Finally, in the fourth step, the 3D shapes are analyzed using contextual knowledge and assigned an *objectness* measure; a measure of the likelihood that a segment corresponds to an object, according to the definition in the introduction.

A. Oversegmentation

The first step (Fig. 2) consists of fusing color and disparity information in a graph-based segmentation algorithm. The goal is to oversegment the scene into a set of coherent regions leveraging on both color, depth, and surface curvature extracted from depth, to disambiguate difficult situations where a single source of information is not enough.

We use the graph-based approach by Felzenszwalb and Huttenlocher [3] to generate coherent regions. Let G_0 be the graph where each vertex $v \in V_0$ is a pixel in a frame. Let RGB, D, and c denote the RGB color value, depth, and curvature at pixel/vertex v, respectively. For details on the computation of curvature, see Sec. III-A.1 at the bottom of next page. Each vertex is connected to 8 other vertices through edges E_0. The weight of the edge (v_1, v_2) is:

$$w_0(v_1,v_2) = \begin{cases} \infty & \text{if } \max(c_1,c_2) > \tau, \\ \alpha\|RGB_1 - RGB_2\| + & \\ (1-\alpha)\|D_1 - D_2\| & \text{if } \min(D_1,D_2) > 0, \\ \|RGB_1 - RGB_2\| & \text{otherwise.} \end{cases} \quad (1)$$

i.e., a weighted sum of the Euclidean distances in the color and disparity spaces, with a threshold τ on curvature to ensure that all high curvature areas correspond to segment boundaries in the resulting oversegmentation. $\alpha \in [0,1]$ is a mixture parameter. Following [3], there is a boundary between two regions R_1 and R_2 if the smallest of their internal variations, $MInt(R_1,R_2)$ is larger than the inter-region variation $Dif(R_1,R_2)$. There is a factor k that governs the preferred segment size – a low k will lead to a fine segmentation, and vice versa. For definitions of these quantities, see [3].

An issue brought to our attention in this step is a consequence of the sensor noise. The alignment between the RGB pixels and depth pixels is not perfect, especially not on the boundaries of objects having disparity values fluctuating between the foreground and the background (Fig. 3). As a result, a multi-modal segmentation can produce noisy regions having outliers with high variation in depth or undersegmenting the scene because of color similarities between the foreground and background. This is leveraged by the introduction of the threshold on curvature, which will enforce segment boundaries in all areas with high depth variation.

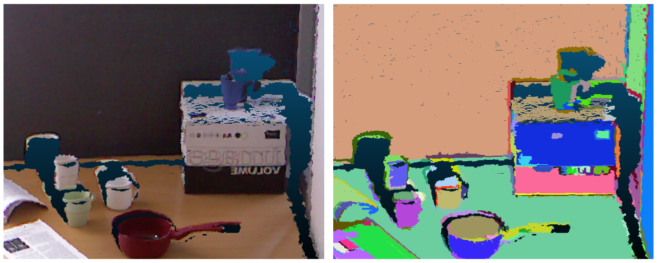

(a) Input Data (b) Segmentation

Fig. 3. Example of noise in the alignment between color and disparity data. Pixels on the boundaries of objects tends to fluctuate between the foreground and the background. As a result the segmentation can generate noisy segments with scattered 3D points.

Due to the local connectivity in G_0, the method can vary widely from oversegmentation to undersegmentation, depending on the value of k. Moreover the merging criteria uses the internal variation of the segments $MInt(R_1)$ that represent the edge having maximum Euclidean distance between two pixels in the Minimum Spanning Tree (MST) of the region and the inter-region variation$Dif(R_1,R_2)$ that is the distance between two pixels connecting different regions. Clearly, as the segments become larger, this merging criteria loses its descriptive power becoming a weak representative feature for the segments. Finally, there is a post-processing parameter n that forces to merge two neighboring segments if one of the two has a size below the value of n. [17] for example set this parameter to a value of 500, a value that in their case represent the size of an object half the size of a mouse. We argue that merging segments only according to their size is not effective and we propose to keep this parameter very low so that it can just remove local noise in the segmentation. Thus, the parameter k is here set so that the scene is oversegmented.

A second segmentation step is introduced (Sec. III-B), which merges the resulting segments into 3D facets, using the same segmentation technique as in the first step described here, but with more long-range image information.

1) Computation of pixel curvature: We here describe how the surface curvature c, used above, is computed.

Normals η are estimated from the depth map D using a real time method based on integral images [15]. The curvature $c(x,y)$ at a pixel (x,y) in the disparity map is defined in terms of the angles between normals in the surrounding pixels. Let the neighborhood pixels be the set of pixels with coordinates $(x \pm \gamma, y \pm \gamma)$.The curvature $c(x,y)$ is then one minus the average over all dot products of normals at opposite neighbor pixels:

$$\begin{aligned} c(x,y) = 1 - \frac{1}{4\sigma}(\eta(x+\gamma,y+\gamma)\cdot\eta(x-\gamma,y-\gamma) + \\ \eta(x+\gamma,y-\gamma)\cdot\eta(x-\gamma,y+\gamma) + \\ \eta(x+\gamma,y)\cdot\eta(x-\gamma,y) \quad + \\ \eta(x,y+\gamma)\cdot\eta(x,y-\gamma) \quad) \end{aligned} \tag{2}$$

where $\eta(x,y)$ is the normal at (x,y), and σ is a scaling parameter. $c(x,y) = 0$ means that the point cloud (x,y) is part of a completely flat surface while increasing values towards 1 will correspond to sharp edges as shown in Fig. 4. We introduce the additional parameter σ to to compensate for effects of the normal computation method [15], where normals are varying less in sparse areas of the point cloud (i.e., surfaces with a large angle to the image plane). We therefore increase σ with point cloud sampling density, creating a curvature estimate which corresponds to the one obtained if points were sampled with equal density everywhere.

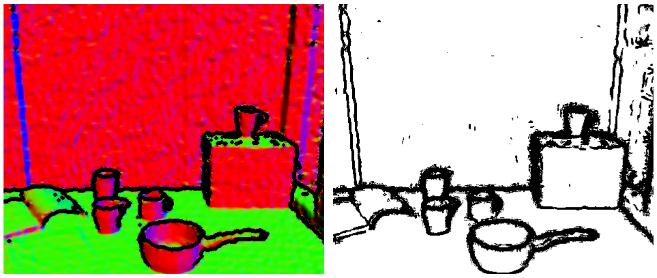
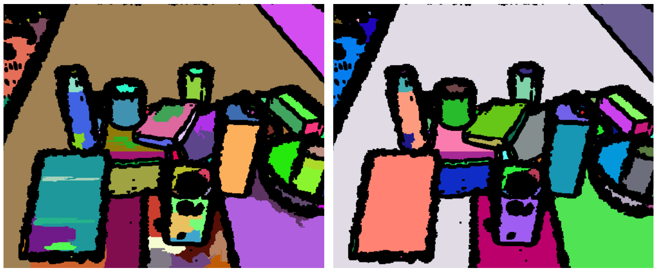

(a) Normal Map (b) Edge Detection

Fig. 4. Example of normal and curvature estimation. In the first figure normal orientations are mapped to RGB space for illustration. In the second image, black pixels show areas with high curvature.

B. Merging Segments into 3D Facets

In the second step, the scene is segmented into 3D surface segments, or 3D facets, which can be considered the smallest physical constituents of objects, if objects are regarded as assemblies of small facets. 3D facets can be defined as areas with coherent normals, i.e., low curvature.

A graph G_1 is defined where each vertex $v \in V_1$ corresponds to a region R resulting from the oversegmentation in step 1 (Sec.III-A). An edge connects a pair of vertices (v_1,v_2) if the two corresponding R_1 and R_2 regions share edges in G_0. All edge weights have a value in the interval $[0,1]$, according to the color and normal histograms of the two connected segments.



(a) Oversegmentation

(b) 3D facet segmentation

Fig. 5. Example of segmentation of 3D surface segments. (a) Original segmentation using local Euclidean distance in color and depth space. (b) 3D surface segments generated by merging regions using histogram intersection of Lab and normal histograms.

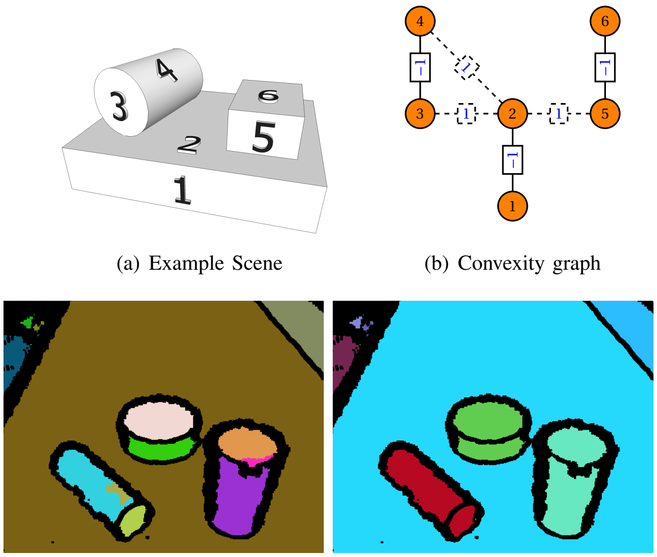
According to experiments (Sec. IV-B) the *CIE L*a*b** (Lab) space was found to be the most efficient for defining color histograms in this segmentation step; let $lab(R)$ be the color histogram of image region R. Normals are defined as in the previous section; let $norm(R)$ be the normal histogram of image region R.

The edge weights of G_1 are defined as:

$$w_1(R_1,R_2) = \\ \omega H_{\cap}(lab(R_1),lab(R_2)) + \\ (1-\omega)H_{\cap}(norm(R_1),norm(R_2)) \qquad (3)$$

where $H_{\cap}(hist(R_1),hist(R_2))$ is the histogram intersection of the histograms of segments R_1 and R_2, and ω is a mixture parameter.

In the original algorithm the segmentation is initialized by considering each pixel as a group and by setting its internal variation to $1/k$. We propose to set the internal variation of each segment to a starting value $c \in [0,1]$ that represent the grade of similarity used to merge different segments. A value of 0 means that the method merges just segment with perfect matching while 1 means that totally dissimilar segments can be merged. The edges in the graph are explored as explained in the previous section and the segments are merged using the same criteria. However when two segments are merged the internal variation of the new generated region is not set to the value of the edge weight that connects the two segments but we calculate the average between the internal variation of the two groups and the edge connecting them. There are two benefits in using this segmentation procedure. First, as the internal variation and the weight of the edges is normalized it is possible to add more features to measure similarity between groups without the need to modify the parameters that control the segmentation. Second, the way the internal variation of merged group is updated allows to have a gradual increase in the similarity needed to merge connected groups while in the original algorithm if two neighboring segments are very similar then it is not possible to merge anymore as the internal variation is very close to 0. The added benefit of this step is exemplified by Fig. 5(b).



(a) Example Scene

(b) Convexity graph

(c) Regions

(d) Convexity analysis

Fig. 6. Example of graph built from a segmented scene. (a) Potential 3D face segmentation result. (b) The graph for this segmentation result, with a node for each segmented region. The computation of edges between regions return a negative value in case of a *ridge* edge and positive for a *valley* edge. Solid edges show the convex connected components. (c) 3D facet segmentation result. (d) The grouping of convex assemblies of 3D facets into 3D shapes.

C. Merging 3D Facets into 3D Shapes

In the third step, 3D facets are assembled into 3D shapes, which are potential *object hypotheses*. We propose to use a convexity criterion: objects are convex assemblies of small surface segments.

The procedure consists of the following steps. First a graph G_c is constructed where each node correspond to a 3D facet. Each node is then connected to other coherent regions that are close in 3D Euclidean space. We derive from the point cloud region R the centroid $C(R)$ and the normal descriptor $\eta(R)$, defined as:

$$\eta(R) = \frac{1}{n}\sum_{1}^{n}\eta(n) \qquad (4)$$

where n is any point of region R.

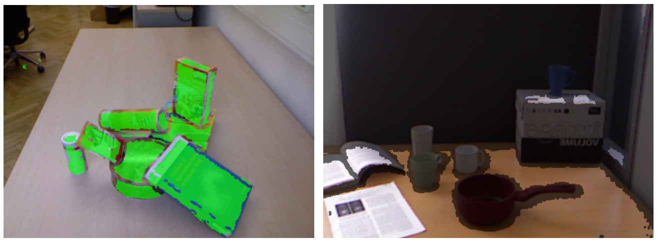
Given two connected regions R_1 and R_2 the value of the edge $w_c(R_1,R_2)$ corresponds to the convexity measure:

$$w_c(R_1,R_2) = (C(R_1) - C(R_2)) \cdot \eta(R_2) \qquad (5)$$

$w_c < 0$ for *ridge* edges, while $w_c > 0$ for *valley* edges. Given the graph shown in Fig. 6(b), the convex structures present in a point cloud can be found using a connected component algorithm to find all nodes connected by negative edges.

D. Objectness measure

We measure the objectness of each 3D shape by exploiting contextual knowledge. The main application of the proposed framework is to formulate valid manipulable object hypotheses therefore objects should: respect law of physics, be of a reasonable size to be manipulated, have a compact shape. Here we explain the features we use to measure the regions.



(a) Filtering by size

(b) Support area detection

Fig. 7. (a) Example of segments filtered by size. Green areas have sizes corresponding to high objectness. (b) Example scene where brighter areas have a high probability to potentially support objects.

a) Size: Manipulable objects should have a reasonable size in order to be handled. Segments with a very big or very small 3D bounding box should be discarder. This allow as to filter out some segments from the possible object list as in Fig. 7(a).

b) Support: An object has to obey the law of physics. Hence, an object is with very high probability found on horizontal surfaces or on top of other supported objects. A usual approach is to find the dominant plain in the scene and then consider objects to be all segments that are connected to it. In this paper we want to go one step further by introducing a recursive approach to find potential objects even if they are not connected to the main planar surface. What we propose is to build a forest of *support tree* given the segmented point cloud computed in the previous steps. Our main assumption is that the robot visual system is oriented horizontally. Therefore we define a directed graph D_g where each node is a point cloud region, nodes are connected if the corresponding regions are connected in $3d$ space and the direction of any edge e connecting the nodes (v_i, v_j) is defined as:

$$e = \begin{cases} e(v_i, v_j), & \text{if } R_i \leq R_j, \\ e(v_j, v_i), & \text{otherwise.} \end{cases} \tag{6}$$

where $R_i \leq R_j$ if R_i vertical position in space is lower than R_j.

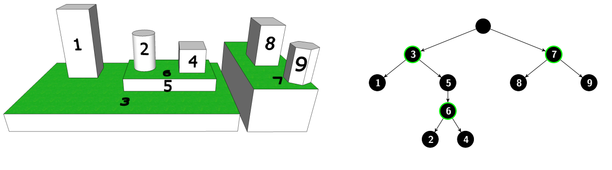
Given that the camera is oriented horizontally a perfect support surface should be characterized by an average normal descriptor (Eq.4) $\eta(RI) = [\eta_x = 0, \eta_y = 1, \eta_z = 0]$. However due to noisy nature of input data such precision can not be obtained. However it is possible to compute the probability $P(S)$ for a point cloud region R to be a support areas as:

$$P(S|\eta(R)) = \eta(RI) \cdot \eta(R) \tag{7}$$

Given any node v and its set of parents V_p in the graph, the probability $P(O)_v$ to be an object hypothesis is then calculated as:

$$P(O)_v = \max(P(S)_p, P(O)_p), \text{with p} \in V_p \tag{8}$$

The computation in Eq. 8 allows to propagate the likelihood to be an object hypothesis to segments that are not support by the main planar surface but are supported by regions with high likelihood to be objects or support regions. Fig. 8 shows a synthetic scenario of stacked objects. It can be seen in Fig. 8(a) the corresponding directed graph. The nodes shown in green have high probability to be support areas.



(a) Scene

(b) Support forest

Fig. 8. Example of how the support forest tree is built. Given a segmented point cloud support segments are found and trees of connected segments are built. Support segments are marked in green.

IV. Experiments

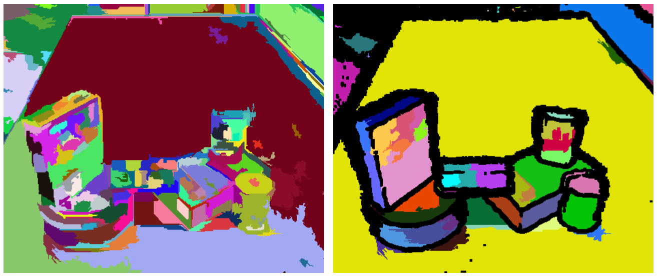
A. Experiments Scenario

The experiments are performed using two datasets, one publicly available [29] that provides table top scenarios and one recorded in house that provides more general indoor scenarios. However the quantitative evaluation are performed only on the publicly available dataset. No qualitative comparison has been performed with the supervised method proposed by [29] as our framework is unsupervised. The purposes of the experiments are four. First we want to show the contribution of the different segmentation step to the final coherent region generation. Second we want to show how a multi-modal approach can disambiguate difficult scenarios where a single source of information is not enough. Third we want to evaluate the accuracy of the method in detecting segments with a high probability to correspond to object hypotheses. And last we want to show how the method cluster regions in object hypotheses given the convexity measurement for edges we described in Sec. III-C.

All experiments on the dataset are performed keeping the same parameter values. The RGB data are smoothed using a Gaussian smoothing filter with standard deviation 0.5 following the settings of the original graph-segmentation paper. The graph segmentation parameters k and n are set to 100 in order to generate oversegmented results; the results are insensitive to the value of these parameters as long as they are much smaller than the total number of pixels in the image. The mixture parameters α and ω are set to 0.5, giving equal weight to all cues. The γ parameter used to compute normals is set to 2. The curvature value threshold τ is set to 0.14, representing an angle of 28 degrees between normals at distance 2γ.

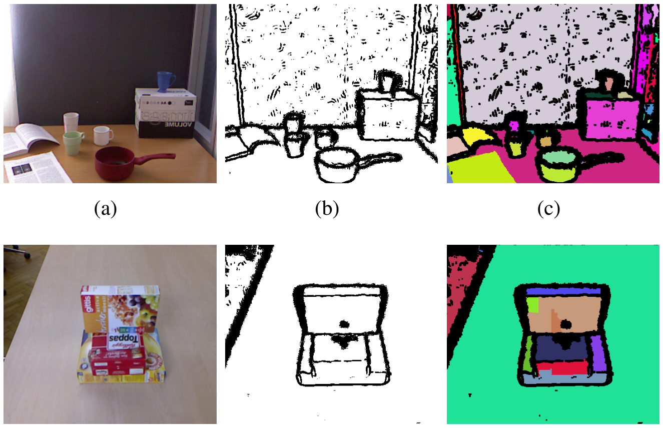
B. Evaluating Oversegmentation

In this section we want to show the contribution of the different sources of information to the segmentation. All experiments are performed using the same parameter scenario. We start with a standard RGB segmentation that shows how coherent regions in a cluttered environment often do not respect real world boundaries (Fig. 9(a)). Second we show the benefit of depth data; a segmentation using RGB-D helps to segment more coherent regions (Fig. 9(b)).



(a) RGB (b) RGB-D

Fig. 9. Example of segmentation post refinement. (a) Oversegmentation from RGB data. Some real boundaries are not respected even if the segmentation parameters are set very low. (b) Oversegmentation from RGB-D using our approach.



(a) (b) (c)

(d) (e) (f)

Fig. 11. Example of difficult scenarios where a source of information is not enough. (a,d) Original image. (b) The paper on top of the table is not detected. (e) The red and yellow boxes are aligned. (c,f) The contribution of color information helps correctly segmenting the paper.

Reversely, depth by itself is often inadequate, the step 1 segmentation benefits from both color and depth. In Fig. 11(c), the method can detect the paper on the table thanks to the contribution of color, in the second example (Fig. 11(f)) the aligned objects can be segmented correctly.

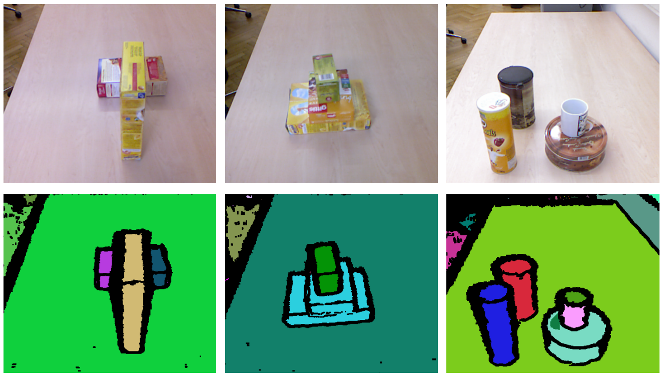
C. Evaluating the Formulation of Object Hypotheses

Fig. 10 shows some results generated by our method and all the intermediate steps. First the oversegmentation (Fig. 10(b)) is merged into 3D facets (Fig. 10(c)). Then the 3D facets are clustered together using convexity (Fig. 10(d)). Finally the method formulates valid object hypotheses by analyzing the 3D coherent shapes, maintaining those have a reasonable manipulable size and are supported directly or indirectly by support areas, as described in Sec. III-D. This is the crucial step to allow a robot to automatically extract object hypotheses.

Therefore we perform a qualitative evaluation of the method on the data set [29] that includes 111 different scenarios of increasing complexity starting from scenes with only 2 objects up to 16. As it is shown in Fig. 12(a) we correctly detect 323 objects out of 430. The other columns in the table show the limitations of the current algorithm. First the method does not deal with occlusion; if the image appearance of an object is cut in half by another one it is going to be segmented in two objects as in Fig. 12(e). If objects are aligned and have similar colors then the algorithm cannot disambiguate the situation and produces an under segmented result (Fig. 12(f)). When the color information dominates and the disparity information is unreliable an object may be over segmented. This is the case in Fig. 12(g) where the high reflectivity property of the object causes over segmentation. We consider an error even if small parts of an object are not clustered in the object hypothesis like in Fig. 12(g). Finally our method does not address concave objects such as cups yet (Fig. 12(g)).

	Correct	Occlusion	Over	Under	Concavity	Total
	323	24	15	35	32	430
%	0.75	0.05	0.035	0.081	0.075	1

(a)

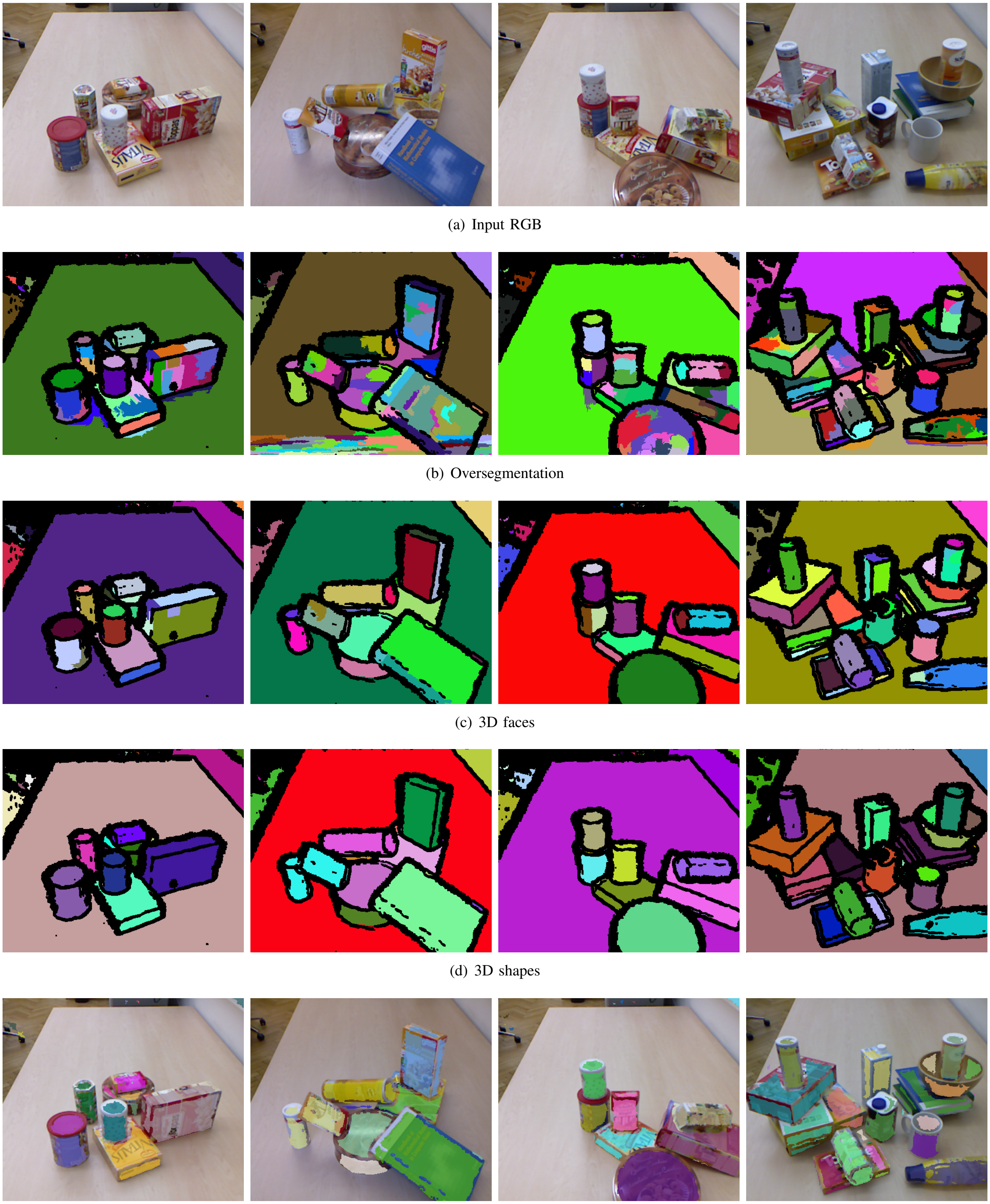


(e) (f) (g)

Fig. 12. Results of the method on an available dataset (a). The second column shows the number of objects correctly classified. Third column shows the number of oversegmented objects. The fourth shows the number of undersegmented objects and the fifth the number of concave objects present in the dataset. Some examples of the limitation are shown. (b) The occluded object is split. (c) Objects are merged becuase of the similarity in color and normal space. (d) Object is oversegmented due to strong color component or reflection of light. The concave object is also oversegmented.

The other columns in the table show the failure cases of the current algorithm. First the method does not deal with occlusion; if the image appearance of an object is cut in half by another one it is going to be segmented in two objects as in Fig. 12(a). Moreover, if objects are aligned and have similar colors then the algorithm cannot disambiguate the situation and produces an under segmented result (Fig. 12(e)). Furthermore, when the color information dominates an object may be over segmented (Fig. 12(f)). Note that our definition of under- or over segmentation is very conservative – we consider it an error even if small parts of an object are not clustered in the object hypothesis, see Fig. 12(g). Finally our method does not address concave objects such as cups yet (Fig. 12(g)).

We believe that we can overcome these issues by adding temporal cues and consistency: when objects are moved it is possible to reason about their state changes to possibly mitigate occlusion. Moreover, over segmentation of object hypotheses can be solved by looking at the local spatial relationship of segments when they are moving.



(a) Input RGB

(b) Oversegmentation

(c) 3D faces

(d) 3D shapes

(e) High objectness

Fig. 10. Segmentation of a cluttered scenes. (a) Examples of input RGB images. (b) Oversegmentations of those examples. (c) 3D faces extracted from those oversegmentations. (d) 3D shapes merged from those 3D facets. (e) 3D shapes with high objectness measure.

V. Conclusions

We propose a general hierarchical graph-based method for segmentation of scenes into regions and estimation of the objectness of these regions. The result is an estimate of the manipulable objects in the scene, which can be used for reasoning about objects and their functionality in manipulation activities. This system is more specific than a general segmentation system, in that it reasons about the physical plausibility of the segmented regions, leading to hypothetic object regions that are likely to correspond to physical entities in the world. Moreover, it is less specific than an object detection system, which can only reason about known object classes; our system is able to generate hypotheses of previously unseen object classes.

Future improvements include other data channels such as sound or thermal cameras.

We also plan to use temporal cues and temporal appearance consistency to improve our method. This can be done in two ways: First, it would be valuable to study whether regions obey the laws of physics in terms of dynamics – that regions follow ballistic trajectories when thrown for example. This can be realized by adding an objectness evaluation procedure on motion and there study the acceleration of regions over time. Regions with a downward acceleration g are coherent with the laws of nature and can be expected to be actual objects. Secondly, temporal cues can improve the clustering of 3D facets, done currently using convexity only. This can be done by using the fact that regions belonging to the same object move together.

Another extension is in terms of the graph segmentation method: The support forest tree based on the normal computation is used to find support areas and regions that are supported directly or indirectly by them. This helps in finding object hypotheses among all those segments with a reasonable manipulable size. However the scene tree can be used to define more contextual relationships between segments in order to reason about object relationships and activity recognition.

The method in the current status is fast but cannot achieve real time performances yet. Most parts of the method are easily parallelized as they rely on very local computation on images or point clouds. The bottle neck of the method is the edge exploration in the graph that requires to explore edges sequentially in ascending order. This is required since the segmentation algorithm is a modified version of Kruskal's algorithm [30] to find a minimum spanning tree in a graph. We plan to approximate this with a parallel minimum spanning tree algorithm, obtaining real time performance, which is required for our robotics applications.

References

[1] R. Unnikrishnan, C. Pantofaru, and M. Hebert. Toward objective evaluation of image segmentation algorithms. *PAMI*, 29(6):929–44, 2007.

[2] J. J. Gibson. *The Ecological Approach to Visual Perception*. Lawrence Erlbaum Associates, 1979.

[3] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 59(2):167–181, 2004.

[4] A. Stein, T. Stepleton, and M. Hebert. Towards unsupervised whole-object segmentation: Combining automated matting with boundary detection. In *CVPR*, 2008.

[5] B. Leibe, A. Leonardis, and B. Schiele. Robust object detection with interleaved categorization and segmentation. *IJCV*, 77(1–3):259–289, 2008.

[6] L. Fei-Fei, R. Fergus, and A. Torralba. *Recognizing and Learning Object Categories: Short course at ICCV*, 2009.

[7] A. Gupta, A. Kembhavi, and L. S. Davis. Observing human-object interactions: Using spatial and functional compatibility for recognition. *PAMI*, 31(10):1775–1789, 2009.

[8] H. Kjellström, J. Romero, and D. Kragic. Visual object-action recognition: Inferring object affordances from human demonstration. *CVIU*, 115(1):81–90, 2011.

[9] H. Grabner, J. Gall, and L. van Gool. What makes a chair a chair? In *CVPR*, 2011.

[10] H. Pirsiavash and D. Ramanan. Detecting activities of daily living in first-person camera views. In *CVPR*, 2012.

[11] A. Pieropan, C. H. Ek, and H. Kjellström. Functional object descriptors for human activity modeling. In *ICRA*, 2013.

[12] J. Shi and J. Malik. Normalized cuts and image segmentation. *PAMI*, 22:888–905, 1997.

[13] M. Grundmann, V. Kwatra, M. Han, and I. Essa. Efficient hierarchical graph-based video segmentation. In *CVPR*, 2010.

[14] A. Abramov, K. Pauwels, J. Papon, F. Worgotter, and B. Dellen. Depth-supported real-time video segmentation with the kinect. In *IEEE Workshop on Applications of Computer Vision*, 2012.

[15] R. B. Rusu D. Holz, S. Holzer and S. Behnke. Real-time plane segmentation using rgb-d cameras. In *RoboCup Symposium*, 2011.

[16] R. Haschke A. Ückermann and H. Ritter. Real-time 3d segmentation of cluttered scenes for robot grasping. In *IEEE-RAS International Conference on Humanoid Robots*, 2012.

[17] A. Karpathy, S. Miller, and L. Fei-Fei. Object discovery in 3d scenes via shape analysis. In *ICRA*, 2013.

[18] S. Srinivasa A. Collet and M. Hebert. Structure discovery in multi-modal data : a region-based approach. In *ICRA*, 2011.

[19] P.F. Felzenszwalb, R.B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *PAMI*, 32(9):1627–1645, 2010.

[20] C. Rother, V. Kolmogorov, and A. Blake. "GrabCut": Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics*, 23(3):309–314, 2004.

[21] S. Bagon, O. Boiman, and M. Irani. What is a good image segment? A unified approach to segment extraction. In *ECCV*, 2008.

[22] A. K. Mishra, Y. Aloimonos, and L. F. Cheong. Active segmentation with fixation. In *ICCV*, 2009.

[23] M. Björkman and D. Kragic. Active 3D scene segmentation and detection of unknown objects. In *ICRA*, 2010.

[24] N. Bergström, C. H. Ek, M. Björkman, and D. Kragic. Scene understanding through interactive perception. In *International Conference on Computer Vision Systems*, 2011.

[25] M. Bar. Visual objects in context. *Nature Reviews Neuroscience*, 5(8):617–629, 2004.

[26] A. Oliva and A. Torralba. The role of context in object recognition. *Trends in Cognitive Sciences*, 11(12):520–527, 2007.

[27] N. Gronau, M. Neta, and M. Bar. Integrated contextual representation for objects' identities and their locations. *Journal of Cognitive Neuroscience*, 20(3):371–88, 2008.

[28] Z. Jia, A. Gallagher, A. Saxena, and T. Tsuhan. 3d-based reasoning with blocks, support, and stability. In *CVPR*, pages 1–8, June 2013.

[29] Object segmentation dataset. Dataset available at `http://www.acin.tuwien.ac.at/?id=289`.

[30] J. B. Kruskal. On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem. *Proceedings of the American Mathematical Society*, 7(1):48–50, February 1956.