Object Partitioning using Local Convexity

Simon Christoph Stein, Markus Schoeler, Jeremie Papon and Florentin Wörgötter
Bernstein Center for Computational Neuroscience (BCCN)
III Physikalisches Institut - Biophysik, Georg-August University of Göttingen
{scstein,mschoeler,jpapon,worgott}@physik3.gwdg.de

Abstract

The problem of how to arrive at an appropriate 3D-segmentation of a scene remains difficult. While current state-of-the-art methods continue to gradually improve in benchmark performance, they also grow more and more complex, for example by incorporating chains of classifiers, which require training on large manually annotated data-sets. As an alternative to this, we present a new, efficient learning- and model-free approach for the segmentation of 3D point clouds into object parts. The algorithm begins by decomposing the scene into an adjacency-graph of surface patches based on a voxel grid. Edges in the graph are then classified as either convex or concave using a novel combination of simple criteria which operate on the local geometry of these patches. This way the graph is divided into locally convex connected subgraphs, which – with high accuracy – represent object parts. Additionally, we propose a novel depth dependent voxel grid to deal with the decreasing point-density at far distances in the point clouds. This improves segmentation, allowing the use of fixed parameters for vastly different scenes. The algorithm is straightforward to implement and requires no training data, while nevertheless producing results that are comparable to state-of-the-art methods which incorporate high-level concepts involving classification, learning and model fitting.

1. Introduction and State-of-the-Art

Segmentation of scenes into objects remains one of the most challenging topics of computer vision despite decades of research. To address this, recent methods often use hierarchies which create a rank order that build bottom-up from small localized superpixels to large-scale regions [19, 1, 3]. As an alternative, researchers have also pursued strictly top-down approaches. These began with coarse segmentations using multiscale sliding window detectors [26], later progressing to finer grained segmentations and detections based on object parts [8, 6]. These two avenues of research led naturally to methods which *combine* bottom-up hierarchy building with top-down object- and part-detectors [2, 22, 9]. While these approaches have yielded quite good results even on complex, varied data sets, they have lost much of the generality of learning-free approaches. In general the most powerful methods to-date use trained classifiers for segmentation [22, 9]. This means they cannot be applied to arbitrary unknown scenes without being re-trained, requiring the acquisition of a new data-set tailored to each test environment.

In this work we investigate model- and learning-free bottom-up segmentation of 3D point clouds captured with RGB-D cameras. In particular, we focus on the partitioning of cluttered scenes into basic *object parts* without the need for training data. As inspiration for a general rule for breaking scenes into elemental parts, we look to psychophysical studies, mostly performed on 2D images, which suggest that the transition between convex and concave image parts might be indicative of the separation between objects and/or their parts [13, 25, 21, 15, 7, 4]. While this feature has been used in machine vision to some degree [10, 17, 20, 24, 11] success has remained limited and more recent studies were forced to combine this feature with additional, often very complex feature constellations to achieve good scene partitioning [20, 24, 11]. It, thus, appears that direct transfer from 2D to 3D of the conventional, geometrically-defined convex-concave transition criterion shown in Fig. 2 A is not possible for achieving good 3D-segmentation. This puzzling observation can best be understood by ways of an example.

Fig. 1 A, left shows two cases, one of which humans will commonly consider as a single object (the left-most structure), while they report the jagged staircase on the right as consisting of three parts. The simple convex-concave criterion will usually fail in such situations because it can produce two substantial errors: On the one hand, it may bind patches across *surface singularities*, one of which is depicted in Fig. 1 A (blue box). On the other hand it may separate objects along *isolated concavities* (Fig. 1 A, red bars). There is a natural trade-off between both errors, making an appropriate segmentation of both cases generally not possi-

ble. Thus, in this study we design a novel 3D-compatible convex-concave separation criterion, which addresses these two problems in a straight-forward way and combine it with a depth dependent transform which helps to reduce the effect of noisy and sparse depth measurements from RGB-D sensors. Together this leads to a remarkably accurate partitioning of real scenes, which can compete with far more complex state-of-the-art methods as shown here by several standard benchmarks. The algorithm is simple and easy to implement and we demonstrate that the resulting segmentations can be understood in human terms as "objects" or "object-parts".

This paper is organized as follow: First, in Section 2 we present our segmentation algorithm and visualize its individual steps. In Section 3 we evaluate our method, benchmark it against other approaches and discuss the results. Finally, Section 4 will summarize our findings. The method source code is freely distributed as part of the Point Cloud Library (PCL)[1].

2. Methods

Our principal goal is to deconstruct a scene into "nameable" object parts without the need for training or classification. As psycho-physical studies suggest that the lowest-level decomposition of objects into parts is closely intertwined with 3D concave/convex relationships, we propose an algorithm, an overview of which is given in Fig. 1, that can reliably identify regions of local convexity in point cloud data.

2.1. Building the Surface-Patch Adjacency Graph

To build a surface patch adjacency graph, we use Voxel Cloud Connectivity Segmentation (VCCS) [18], a recent method which over-segments 3D point cloud data into patches, known as supervoxels (a 3D analog of superpixels). VCCS uses a local region growing variant of k-means clustering to generate individual supervoxels $\vec{p}_i = (\vec{x}_i, \vec{n}_i, N_i)$, with centroid $\vec{x}_i$, normal vector $\vec{n}_i$, and edges to adjacent supervoxels $e \in N_i$.

Supervoxels maintain adjacency relations in voxelized 3D space (specifically, 26-adjacency). The adjacency graph of supervoxels (and the underlying voxels) is maintained efficiently within the octree by searching for neighboring leaves in the voxel grid, where R_{voxel} specifies the octree leaf resolution. This adjacency graph is used for both the region growing to generate the supervoxels as well as determining adjacency of the resulting supervoxels themselves. Supervoxels are grown from a set of seed points distributed evenly in space on a grid with resolution R_{seed}. Expansion from the seed points is governed by a distance measure calculated in a feature space consisting of spatial extent, color, and normals. In this work, as we are interested only in geometric features, we only use the spatial weight $w_s = 1$ and the normal direction weight $w_n = 4$. Additionally, we have extended the VCCS algorithm by modifying how the surface normals are calculated. There are two changes: First, rather than using a simple radius-search, we use the adjacency graph to decide which nearby points are included in the normal calculation (specifically, we use the neighbors, and the neighbors-neighbors). Secondly, we use the proba-

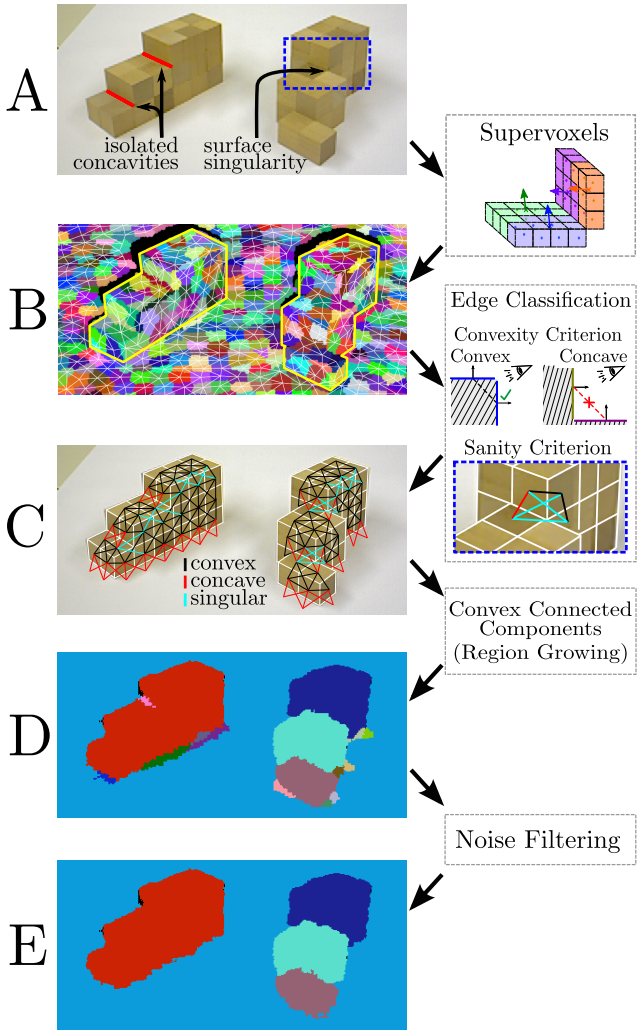


Figure 1. Flow diagram of the segmentation algorithm. **A)** RGB images corresponding to the point clouds of the scene. The red lines show two isolated concavities. The blue box shows an area with a surface singularity. **B)** Supervoxel adjacency graph. **C)** Model depicting the classified graph. Black lines denote convex connections, red lines concave ones and turquois lines singular connections (those, where two patches are connected only in a single point). **D)** Segmentation result; object labels are shown by different colors. **E)** Final result after noise filtering. The right column illustrates the supervoxel patches and the convexity and sanity criteria used for edge classification.

[1] http://www.pointclouds.org

bilistic normal method of Boulch and Marlet [5]. The combination of these two changes makes normals significantly sharper, resulting in supervoxels which conform better to sharp edges.

2.2. Locally Convex Connected Patches (LCCP)

Next we segment the supervoxel adjacency graph by classifying whether the connection $e = (\vec{p}_i, \vec{p}_j)$ between two supervoxels is convex (=valid) or concave (=invalid).

Extended Convexity Criterion (CC) Consider two adjacent supervoxels with centroids at the positions $\vec{x}_1, \vec{x}_2$ and normals $\vec{n}_1, \vec{n}_2$. Whether the connection between these is convex or concave can be inferred from the relation of the surface normals to the vector joining their centroids.

The angle of the normals to the vector $\vec{d} = \vec{x}_1 - \vec{x}_2$ joining the centroids can be calculated using the identity for the dot product $\vec{a}\cdot\vec{b} = |\vec{a}|\cdot|\vec{b}|\cdot\cos(\alpha)$ with $\alpha = \measuredangle(\vec{a}, \vec{b})$. For *convex* connections, α_1 is smaller than α_2 (see Fig. 2 A). This can be expressed as:

$$\alpha_1 < \alpha_2 \Rightarrow \cos(\alpha_1) - \cos(\alpha_2) > 0 \Leftrightarrow \vec{n_1}\cdot\hat{d} - \vec{n_2}\cdot\hat{d} > 0,$$

where $\hat{d} = \frac{\vec{x_1}-\vec{x_2}}{||\vec{x_1}-\vec{x_2}||}$. Similarly, for a *concave* connection we get:

$$\alpha_1 > \alpha_2 \Leftrightarrow \vec{n_1}\cdot\hat{d} - \vec{n_2}\cdot\hat{d} < 0.$$

Note that these operations are commutative, thus the choice of which patch is $\vec{x}_1$, does not change the result. Also the criterion is still valid if the $\vec{x}_i$ are displaced, as long as they stay within the surface.

To compensate for noise in the RGB-D data, a bias is introduced to treat concave connections with very similar normals, that is

$$\beta = \measuredangle(\vec{n}_1, \vec{n}_2) = |\alpha_1 - \alpha_2| = \cos^{-1}(\vec{n_1}\cdot\vec{n_2}) < \beta_{\text{Thresh}}\ ,$$

as convex, since those usually represent flat surfaces. Depending on the value of the *concavity tolerance threshold* β_{Thresh}, concave surfaces with low curvature are seen as convex and thus merged in the segmentation. This behavior may be desired to ignore small concavities. We set:

$$\text{CC}_b(\vec{p}_i, \vec{p}_j) := \begin{cases} \text{true} & (\vec{n_1} - \vec{n_2})\cdot\hat{d} > 0 \ \vee\ (\beta < \beta_{\text{Thresh}}) \\ \text{false} & \text{otherwise.} \end{cases} \tag{1}$$

where the variable CC_b defines the *basic convexity criterion*. However, local errors in the feature estimation caused by noise in the data can propagate very easily, potentially leading to errors in the resulting segmentation. This also makes the recognition of small concavities harder, as subtle features are more sensitive to noise. To improve on this we – finally – also include neighborhood information in the classification of edges: For a convex edge $e = (\vec{p}_i, \vec{p}_j)$, we require that there exists a common neighbor $\vec{p}_c$ of $\vec{p}_i$ and $\vec{p}_j$ that has a convex connection to both.

Thus we define *extended convexity* CC_e:

$$\begin{aligned}\text{CC}_e(\vec{p}_i, \vec{p}_j) = \text{CC}_b(\vec{p}_i, \vec{p}_j) \wedge \text{CC}_b(\vec{p}_i, \vec{p}_c) \\ \wedge \text{CC}_b(\vec{p}_j, \vec{p}_c)\end{aligned} \tag{2}$$

With extended convexity, more evidence is necessary for a connection to be labeled as convex. Our implementation is based on the idea proposed by Moosmann [16]. In some cases results are already satisfactory even without using this type of neighborhood information. Thus, in the results section we will always denote whether this is used or not.

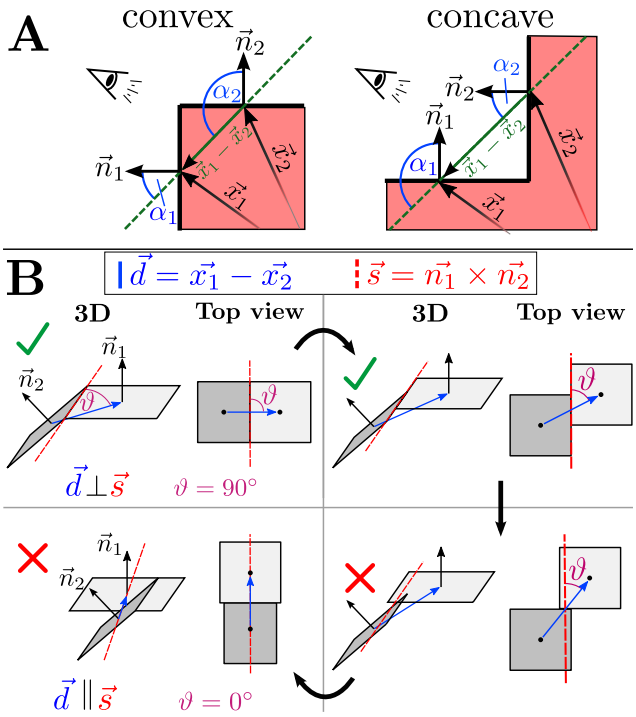


Figure 2. **A)** Illustration of measures used in the basic convexity criterion. **B)** Illustration of the sanity criterion for decreasing values of ϑ. Singular connections are characterized by small values of ϑ.

Sanity criterion (SC): As pointed out in the Introduction when discussing the problem of *surface-singularities*, in such cases, the classification of the connection of two supervoxels into convex or concave does not make sense. Hence, if the surface is discontinuous this is evidence for a geometric boundary. This means that the corresponding (originally potentially convex) connections should be identified and invalidated. To do this, we use the vector $\vec{d}$ which connects the centroids and the direction of intersection $\vec{s} = \vec{n}_1 \times \vec{n}_2$ of the planes approximating the supervoxel surface. As illustrated in Fig. 2 B, singular configurations can be identified by looking at the angle ϑ between $\vec{d}$ and

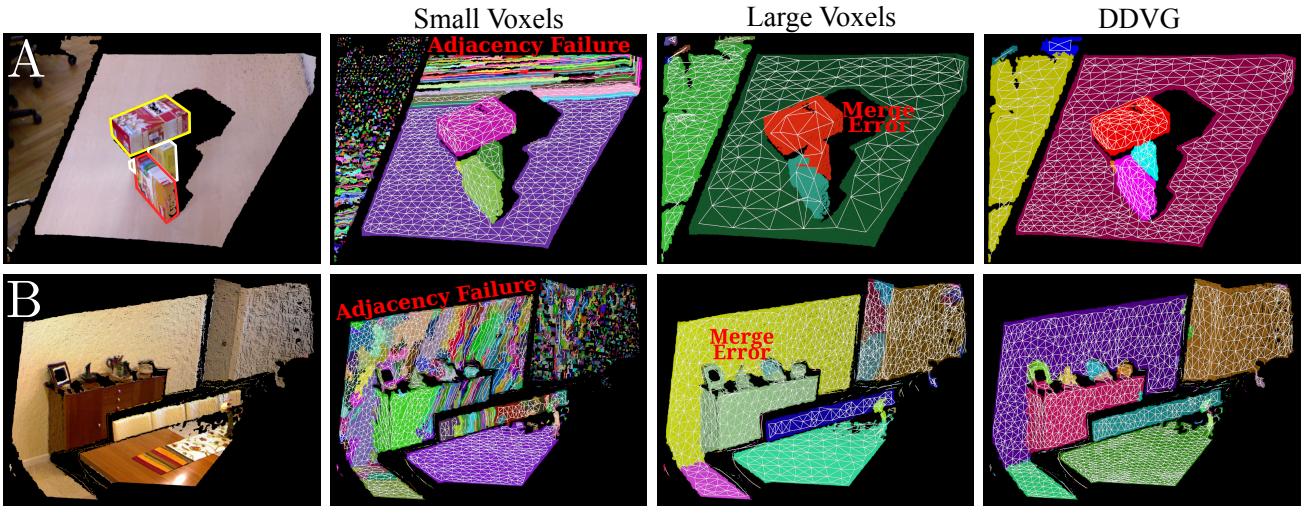


Figure 3. Two example point clouds **(A,B, left)** showing the need for the Depth Dependent Voxel Grid. For better visibility outlines have been drawn around the boxes in A. Using *Small Voxels* objects close to the camera can be segmented, but adjacency breaks down as the depth increases and the point density decreases. Using *Large Voxels* corrects the adjacency graph in the background, but leads to objects being merged in the foreground due to the coarse resolution. Using *DDVG*, the scale of the voxels gradually increases with distance from the camera – adapting to the increased noise level and lower point density – consequently adjacency is maintained and the segmentation of scenes with large depth variance is possible using fixed parameters. Note, the flat rug on the table in B does not differ enough from the table's surface and cannot be segmented by any purely depth dependent method.

$\vec{s}$. For two supervoxels sharing one side of their boundary, the two directions are orthogonal ($\vartheta = 90°$). If the directions are parallel ($\vartheta = 0°$), the patches are only connected in a single point and the situation is singular. Because the orientation of $\vec{s}$ is arbitrary, we define ϑ to be the minimum angle between the two directions, that is:

$$\begin{aligned}\vartheta(\vec{p}_1, \vec{p}_2) &= \min(\measuredangle(\vec{d}, \vec{s}), \measuredangle(\vec{d}, -\vec{s})) \\ &= \min(\measuredangle(\vec{d}, \vec{s}), 180° - \measuredangle(\vec{d}, \vec{s}))\end{aligned} \tag{3}$$

Singular configurations occur for small values of ϑ and can thus be handled by introducing the threshold $\vartheta_{\text{Thresh}}$. For $\vartheta < \vartheta_{\text{Thresh}}$ the connection must be invalidated. Similar to the convexity criterion, this condition has to be relaxed for supervoxels with very similar normals, to compensate for sensor noise. This is done by setting $\vartheta_{\text{Thresh}}(\measuredangle(\vec{n}_1, \vec{n}_2))$ to a sigmoid function (a softened step function) of the angle between normals:

$$\vartheta_{\text{Thresh}}(\beta) = \vartheta^{\text{max}}_{\text{Thresh}} \cdot \left(1 + \exp\left[-a \cdot (\beta - \beta_{\text{off}})\right]\right)^{-1}, \tag{4}$$

where $\beta = \measuredangle(\vec{n}_1, \vec{n}_2)$ is the angle between normals. We use the experimentally derived values $\vartheta^{\text{max}}_{\text{Thresh}} = 60°$, $\beta_{\text{off}} = 25°$ and $a = 0.25$.

The sanity criterion SC is then defined as

$$\text{SC}(\vec{p}_i, \vec{p}_j) := \begin{cases} \text{true} & \vartheta(\vec{p}_i, \vec{p}_j) > \vartheta_{\text{Thresh}}(\beta(\vec{n}_1, \vec{n}_2)) \\ \text{false} & \text{otherwise} \end{cases} \tag{5}$$

We shall denote convex edges, that is, those which satisfy both 1 and 5, as satisfying

$$\text{conv}(\vec{p}_i, \vec{p}_j) = \text{CC}_{b,e}(\vec{p}_i, \vec{p}_j) \wedge \text{SC}(\vec{p}_i, \vec{p}_j)\ . \tag{6}$$

Region Growing: The second problem discussed in the Introduction, the danger of splitting objects along *isolated concavities*, can be addressed in the next step. For this we need to find all *clusters* of supervoxels, which belong to the same subgraph of valid convex connected edges. This is accomplished using a region growing process: First, an arbitrary seed supervoxel is chosen and labeled. This label is then propagated over the graph with a depth search that is only allowed to grow over convex edges. Once no new supervoxel can be assigned to the segment, we choose a new seed supervoxel that has not been labeled and propagate the new label as before, repeating the process until all supervoxels have been labeled. Note that all our criteria are commutative, so the output of the region growing does not depend on the choice of the seeds.

Noise Filtering: Noisy surface normals (predominantly present at boundaries) can lead the convexity classification to fail and wrongfully split connected surfaces. Because these noisy patches are usually small, they can be filtered out in a post-processing step. For every segment of the segmentation, we check if it contains at least n_{filter} supervoxels. If a segment's size is smaller or equal to the filter size, we merge it with the neighboring segment with the greatest

size. The filtering continues until no segments (that have neighbors) smaller than n_{filter} are present in the image. We use $n_{\text{filter}} = 3$ for all results presented in this work.

2.3. Depth Dependent Voxel Grid (DDVG)

So far we have described the main algorithm for segmentation. Next we will introduce a generally applicable depth transform, which improves this specific analysis but can be used for all types of image analyses using algorithms with a fixed scale of observation on RGB-D data from a single RGB-D camera. In our case, we address shortcomings of the voxel grid VCCS is based on.

It is evident that observations from a single RGB-D camera have a significant drawback - the point density, and thus available detail of the scene geometry, falls rapidly with increasing distance from the camera. In addition, the levels of both quantization and noise grow quadratically [23, 12], leading to a further degradation in the quality of geometric features. This change in point density with depth creates a tradeoff between capturing small-scale detail in the foreground (using small voxels) and avoiding noise in the background (using large voxels). This is a general problem which occurs in all algorithms working with a fixed scale (for example a radius search) on point clouds created from a single view.

We propose to compensate for the loss of point density and quantization with increasing depth z by transforming the points into a skewed space using the transformation T : $(x, y, z) \rightarrow (x', y', z')$ with

$$x' = x/z, \quad y' = y/z, \quad z' = \log(z) \tag{7}$$

The division of the x and y coordinates by z reverses the perspective transformation, equalizing the point density in the x-y-plane. Transforming the z coordinate helps to deal with the effects of depth quantization by compressing points as depth increases. It is easy to show that the transformation has the following property:

$$\frac{\partial x'}{\partial x} = \frac{\partial y'}{\partial y} = \frac{\partial z'}{\partial z} = \frac{1}{z} \tag{8}$$

Because the derivatives are equal, the local coordinate frame is stretched equally along all axes by the transformations. The important thing about this property is, that small cubic voxels are still cubic after the transformation. This leaves the geometry of space basically untouched in the foreground (if the voxel size is chosen sufficiently small), while voxels in the background are skewed and grow, to compensate for reduced amount of detail available in the data.

Rather than transforming the clouds back and forth, we instead transform the bins of the octree itself, creating an octree where bin volume (and thus, voxel size) effectively increases with distance from the camera viewpoint. Doing this directly within the octree allows us to determine adjacency as before (neighboring bins), even though distance between neighboring voxels increases with distance from the camera. Fig. 3 illustrates this advantageous effect of this transformation on the segmentation.

3. Evaluation

In the following sections we present qualitative results on some sample images which demonstrate how we segment into natural (nameable) parts. In addition to these qualitative results, we also provide quantitative results which compare to state-of-the-art methods on the *NYU Indoor Dataset*[22] and *Object Segmentation Database*[20]. We compare segments against ground truth using three standard measures: *Weighted Overlap* (WOv), which is a summary measure proposed by Silberman *et al.* [22], as well as *false negative* (fn) and *false positive* (fp) scores from [24] and *over-* (F_{os}) and *under-segmentation* (F_{us}) from [20].

3.1. Object Segmentation Database (OSD)

The *Object Segmentation Database* (OSD-v0.2) was proposed by Richtsfeld *et al.*[20] in 2012. It consists of 111 cluttered scenes of objects on a table, taken with close proximity to the pictured objects. The scenes contain multiple objects, which have mostly box-like or cylindrical shape, with partial and full occlusions and heavy clutter in 2D as well as 3D. Importantly, most objects in the data set are *simple*, that is, consist of only a single part. This makes the ground-truth data relatively non-ambiguous.

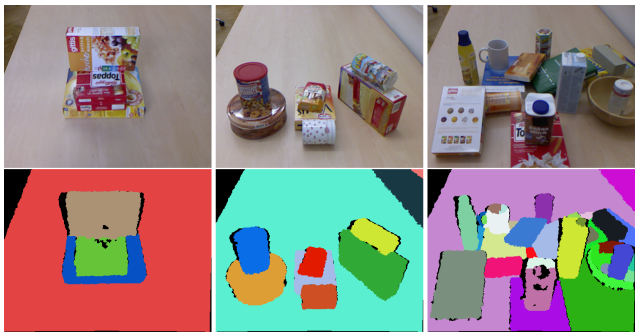


Figure 4. Example results for the OSD dataset. Points beyond a distance of 2m were cropped for visualization. Parameters: $R_{voxel} = 0.005$, $R_{seed} = 0.02$, $\beta_{\text{Thresh}} = 10°$.

The qualitative examples (Fig. 4) show that our algorithm performs very well in the segmentation of these cluttered scenes. The object separation can be intuitively understood: all objects present in the scenes are separated by concave boundaries, i.e. a line connecting neighboring surfaces of two different objects always travels through “air”. This is also true for the boundary between an object and the supporting surface. As a consequence, objects that have

Method	Learned Features	WOv	fp		fn		F_{os}	F_{us}
		Mean	Mean	SD	Mean	SD	Mean	Mean
LCCP	NO LEARNING	88.7%	4.8%	2.6%	8.3%	8.7%	7.4%	4.7%
Richtsfeld *et al.* [20]	RGB-D + Texture + Geometry	-	-	-	-	-	4.5%	7.9%
Ückermann *et al.* [24]	NO LEARNING	-	1.9%	3.3%	7.8%	7.3%	-	-

Table 1. Comparison of different segmentation methods on the OSD dataset using weighted overlap *WOv* (the higher, the better), false positives f_p, false negatives f_n, as well as over- and under-segmentation F_{os} and F_{us} (the lower, the better). LCCP results were produced with voxel resolution $R_{voxel} = 0.005$, seed resolution $R_{seed} = 0.02$ and concavity tolerance angle $\beta_{\text{Tresh}} = 10°$.

a convex shape are correctly captured as one segment and separated from the other objects. Hollow objects (bowls, cups etc.) can be observed to show multiple segments inside, because the orientation of surface normals changes strongly on these concave surfaces. Despite most objects being simple, some objects have meaningful parts, such as handles or bottle caps, which are segmented by our algorithm.

The quantitative results (Table 1) demonstrate that our approach is able to compete with state-of-the-art methods in the task of segmenting cluttered scenes with 'single-part' objects. Compared to the learning-based method from [20] we achieve better object separation (F_{us}), but higher over-segmentation error (F_{os}). The latter is because we sometimes detect object parts (*e.g.* handles) and we do not utilize model fitting, which helps against noise. Comparing to the learning-free method of Ückermann *et al.* [24] we obtain results within a standard deviation. Note that our actual goal is to partition complex objects into parts, which is dissimilar from the other two methods.

3.2. NYU Indoor Dataset (NYU)

The *NYU Indoor Dataset* (NYUv2) from Silberman *et al.*[22] is a large and complex dataset, consisting of 1449 scenes with realistic cluttered conditions. The images are divided into a training and testing set of 795 and 654 images, respectively. In order to compare against other methods we only evaluated on the testing images, even though our method does not require a training set. The distance of objects from the camera is generally quite large in the dataset (considering the operational range of the Kinect camera), resulting in large depth quantization artifacts and few data for many objects. Furthermore, depth is often missing for significant portions of an image, due to various limitations of the Kinect sensor (e.g. reflective, transparent surfaces). To correct for this, the creators provide depth images enhanced by a hole filling algorithm (smoothdepth), which tries to estimate depth for missing areas based on the scheme from Levin *et al.*[14].

Example scenes in Fig. 5 show that the general object separation is still very good, which is expected from the results presented in the previous section. In contrast to the simple objects from the OSD dataset, however, in the NYU dataset the objects of interest are complex objects from various aspects of everyday life. As a consequence, these are mostly composed of multiple parts and thus our algorithm reveals interesting partitions. To give a few examples: In scene (A) the cupboard is partitioned into the top plate and its various drawers and the toilet is segmented into the flushing tank, the seat and the base. Scene (B) presents how a human is partitioned into hand, arm-bed, upper/lower part of the body and legs. For the sofas in scene (D) we get the two arm-rests, the back-rest and the seating. Note that in these cases the segments represent "nameable regions", which is also the case for many other segments (we discuss this further in Section 3.3). It is evident that the ground truth data will then disagree with our labeling, which results in unjustified errors. For example, in scene (A) the ground truth considers the sink to be an important part, while the drawers are not labeled individually. Despite this disagreement, we do not consider either of the labelings wrong in general, but see them as different views on the data which are in many cases equally justifiable.

Because we designed our algorithm to detect object parts defined only by geometry, very thin objects like the posters on the background wall in scene (C) are not recognizable. Also the challenging data quality sometimes leads the part partitioning to fail, as seen for the human in scene (E). To show how well the general idea of part partitioning using concave boundaries works, we present results for a complex object with high data quality in the next section.

The quantitative results (Table 2)[2] show that our algorithm is able to produce good results on the challenging dataset. Despite being much simpler and without requiring learning on human annotated ground-truth, we compete with the approach from [22] when only depth information is used. Additionally, we still achieve 93% of their score when comparing against the more complex feature spaces used in conjunction with learning-based algorithms. We should emphasize that our competitors do not aim for object parts but rather for "whole object" detections, specifically, those whole objects learned from this particular annotated ground truth. Conversely, our method establishes a general rule for object-ness that does not depend on this particular dataset, nor on the whims of a particular human annotator.

[2]Updated results for [22] are available at http://cs.nyu.edu/~silberman/datasets/nyu_depth_v2.html.

Figure 5. Example results for scenes from the NYU dataset using unsmoothed depth. Black areas indicate missing depth. Top row: rgb images. Mid. row: segmentation result. Bottom row: ground truth. Parameters A-C: $R_{voxel} = 0.0075$, $R_{seed} = 0.03$ and $\beta_{\text{Thresh}} = 8°$. Parameters D-E: $R_{voxel} = 0.01$, $R_{seed} = 0.04$ and $\beta_{\text{Thresh}} = 10°$ (identical to quantitative results, see Tab. 2).

Method	Learned Features	Depth Data	WOv
LCCP	NO LEARNING	depth	53.6%
	NO LEARNING	smoothdepth	53.8%
LCCP + ext. convexity	NO LEARNING	smoothdepth	57.6%
Silberman *et al.*[22]	RGB	-	50.3%
	Depth	both	53.7%
	RGB-D	both	60.1%
	RGB-D + Support + Structure classes	both	61.1%
Gupta *et al.*[9]	gPb-ucm Gradients (from [3])	-	55.0%
	gPb-ucm + Depth + Concavity Gradients	both	62.0%

Table 2. Comparison of different segmentation methods on the NYU dataset using weighted overlap *WOv*. LCCP results were produced with voxel size $R_{voxel} = 0.01$, seed size $R_{seed} = 0.04$ and concavity tolerance angle $\beta_{\text{Tresh}} = 10°$.

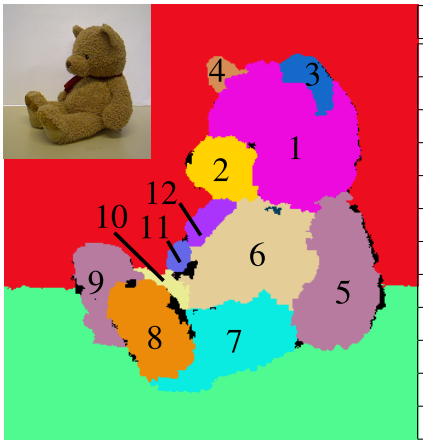
The average runtime was 470 ms per scene using an Intel Core i7 3.2 GHz processor with 95% spent computing the supervoxels. Note that supervoxels can be parallelized using GPUs, which should lead to a 10-fold speed-up.

3.3. Partitioning of Higher Quality Data

Since the NYU dataset and the OSD set do not provide close-ups on complex objects, we present a qualitative example in Fig. 6 which shows results on higher quality data. It is easy to see that our algorithm segments the image into meaningful parts, although we have not learned what constitutes a part. In particular, it is apparent that our algorithm partitions the object into parts which are easily "human-nameable". We should emphasize that the partitions in Fig. 5 A-C and in Fig. 6 were created using the proposed depth dependent voxel grid with the same, fixed parameters. This shows the applicability of our algorithm to disparate content without parameter tuning.

4. Conclusion

In this work we presented and evaluated a novel model- and learning-free bottom-up segmentation algorithm operating on 3D point clouds. Although our algorithm aims to partition parts in contrast to objects, we achieved state-of-the-art results on two well-known benchmarks, the *Object Segmentation Database* and the *NYU Indoor Dataset*. The latter being especially interesting for two reasons: First, all published results require learning (necessitating annotated ground-truth) as well as color information. Our approach, on the contrary, is the first published algorithm which requires neither. Secondly, ground-truth arbitrarily partitions the scene into "full" objects, thus segmenting objects into their parts will decrease overall scores. In spite of all this, we obtain results which are comparable to the state-of-the-art. Finally, we should emphasize that being a learning-free method has many advantages, the most important of



ID	name
1	head
2	snout
3	left ear
4	right ear
5	left arm
6	belly
7	left leg
8	left foot
9	right foot
10	right leg
11	right arm
12	tie

Figure 6. One example scene showing a teddy-bear and it's partitioning. Segments correspond to meaningful and nameable body parts.

which is that there is no need to create new training data and accompanying annotated ground truth images. Not only does this mean that the method is directly applicable as the first step in an automated bootstrapping process, it also enables use on arbitrary unknown scenes (made possible by our depth dependent grid) or scenes acquired by new devices (e.g. Kinect 2.0, or laser scanners).

Acknowledgments

The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 (Specific Programme Cooperation, Theme 3, Information and Communication Technologies) under grant agreement no. 600578, ACAT.

References

[1] N. Ahuja and S. Todorovic. Connected segmentation tree; a joint representation of region layout and hierarchy. In *CVPR*, pages 1–8, 2008. 1

[2] P. Arbelaez, B. Hariharan, C. Gu, S. Gupta, L. Bourdev, and J. Malik. Semantic segmentation using regions and parts. In *CVPR*, pages 3378–3385, 2012. 1

[3] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Trans. PAMI*, 33(5):898–916, 2011. 1, 7

[4] M. Bertamini and J. Wagemans. Processing convexity and concavity along a 2-D contour: figure-ground, structural shape, and attention. *Psychonomic Bulletin & Review*, 20(2):191–207, 2013. 1

[5] A. Boulch and R. Marlet. Fast and robust normal estimation for point clouds with sharp features. *Computer Graphics Forum*, 31(5):1765–1774, 2012. 3

[6] L. Bourdev and J. Malik. Poselets: Body part detectors trained using 3D human pose annotations. In *ICCV*, pages 1365–1372, 2009. 1

[7] A. D. Cate and M. Behrmann. Perceiving parts and shapes from concave surfaces. *Attention, Perception, & Psychophysics*, 72(1):153–167, 2010. 1

[8] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Trans. PAMI*, 32(9):1627–1645, 2010. 1

[9] S. Gupta, P. Arbelaez, and J. Malik. Perceptual organization and recognition of indoor scenes from RGB-D images. In *CVPR*, pages 564–571, 2013. 1, 7

[10] R. Hoffman and A. K. Jain. Segmentation and classification of range images. *IEEE Trans. PAMI*, 9(5):608–620, 1987. 1

[11] A. Karpathy, S. Miller, and L. Fei-Fei. Object discovery in 3D scenes via shape analysis. In *ICRA*, pages 2088–2095, 2013. 1

[12] K. Khoshelham and S. O. Elberink. Accuracy and resolution of kinect depth data for indoor mapping applications. *Sensors*, 12(2):1437–1454, 2012. 5

[13] J. J. Koenderink. What does the occluding contour tell us about solid shape? *Perception*, 13:321–330, 1984. 1

[14] A. Levin, D. Lischinski, and Y. Weiss. Colorization using optimization. *ACM Trans. Graph.*, 23(3):689–694, 2004. 6

[15] T. Matsuno and M. Tomonaga. An advantage for concavities in shape perception by chimpanzees (pan troglodytes). *Behavioural Processes*, 75(3):253–258, 2007. 1

[16] F. Moosmann. *Interlacing Self-Localization, Moving Object Tracking and Mapping for 3D Range Sensors*. PhD thesis, Karlsruher Institut für Technologie (KIT), 2013. 3

[17] F. Moosmann, O. Pink, and C. Stiller. Segmentation of 3D lidar data in non-flat urban environments using a local convexity criterion. In *Intelligent Vehicles Symposium*, pages 215–220, 2009. 1

[18] J. Papon, A. Abramov, M. Schoeler, and F. Wörgötter. Voxel cloud connectivity segmentation - supervoxels for point clouds. In *CVPR*, pages 2027–2034, 2013. 2

[19] X. Ren and J. Malik. Learning a classification model for segmentation. In *ICCV*, pages 10–17 vol.1, 2003. 1

[20] A. Richtsfeld, T. Morwald, J. Prankl, M. Zillich, and M. Vincze. Segmentation of unknown objects in indoor environments. In *IROS*, pages 4791–4796, 2012. 1, 5, 6

[21] P. L. Rosin. Shape partitioning by convexity. *IEEE Trans. SMC, Part A: Systems and Humans*, 30:202–210, 2000. 1

[22] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from RGBD images. In *ECCV*, pages 746–760, 2012. 1, 5, 6, 7

[23] J. Smisek, M. Jancosek, and T. Pajdla. 3D with kinect. In *ICCV Workshops*, pages 1154–1160, 2011. 5

[24] A. Ückermann, R. Haschke, and H. Ritter. Real-time 3D segmentation of cluttered scenes for robot grasping. In *Humanoids*, 2012. 1, 5, 6

[25] L. M. Vaina and S. D. Zlateva. The largest convex patches: A boundary-based method for obtaining object parts. *Biol. Cybern.*, 62(3):225–236, 1990. 1

[26] P. Viola and M. Jones. Robust real-time face detection. *IJCV*, 57(2):137–154, 2004. 1