Sweep Prior, from 2.5D to 3D

Rufei Jiang
Shanghai Jiao Tong University
Shanghai, P.R.China
jrfjrf@sjtu.edu.cn

Weijun Zhang
Shanghai Jiao Tong University
Shanghai, P.R.China
zhangweijun@sjtu.edu.cn

Abstract

Is vision a process, or just a set of individual tasks? Though it's common to ignore this question and just jump into one specific task, researchers did believe vision is a process, and proposed a representation framework to tackle vision problems four decades ago.

This view may be right or not, but we gain a lot of inspiration from it. By integrating vision tasks into a similar three phases framework, we argue that the second phase of vision is obtaining 3D shapes and the positions of objects, even when the objects are not known. We assign two tasks for second phase, 2.5D segmentation and shape completion of unknown objects, and find those two tasks are tightly related to each other. In this work, we introduce our framework first, then propose a prior for the second phase. This prior, called sweep prior, is a simple assumption that all 3D objects are the decomposition of parts constructed by cross section sweep. Based on sweep prior, we demonstrate a theory and an algorithm to handle those two tasks of second phase. Our algorithm can only be applied to simple 3D scenes currently, so the limitations of our algorithm are discussed at last.

1. Introduction

Vision problems are often divided into independent tasks to tackle. Using a computational approach, researchers confirmed that vision tasks have a relationship, and among these tasks surface normal estimation supplies the most useful information for other relevant tasks [22].

Surprisingly, this conclusion purely derived from computation agrees with Marr's theory four decades ago. In his book [10], Marr proposed a representation framework, concluding that human perception goes in three steps: from raw image to primal sketch; from primal sketch to 2.5D sketch; form 2.5D sketch to 3D model. 2.5D sketch is defined as local surface orientation and discontinuities in depth and in surface orientation, which is exactly surface normal. Marr didn't show a clear method how to get these representations, but his idea of framework inspires us a lot: maybe vision tasks can go beyond pair relationship and be arranged into a unified framework.

Following Marr's theory, we select some typical tasks and roughly divide them into three phases: depth estimation; 2.5D segmentation and shape completion of unknown objects; semantic and instance segmentation. Recovering depth is the first key phase, because it provides us geometry representation from variant texture and lighting condition. Wu *et al*. [21] finds that for shape completion, the use of 2.5D sketch provides better domain transfer ability compared to other end to end deep learning methods.

The decomposition criteria of the second and third phase is based on an observation: humans can understand 3D environments before knowing object classes. So we argue that semantic cognition is the last step of 3D perception, and vision works as an inverse graphics process right after getting 2.5D sketch: extract the full model of objects and their positions. To achieve this, we should deal with 2.5D segmentation to segment objects apart, then handle shape completion to recover objects' full 3D shape. Importantly, both tasks must be done when objects are not known. In this work, we concentrate on the second phase, from 2.5D to 3D, and propose a simple prior to solve these two tasks together.

Let's start with shape completion. Shape completion is considered to be an ill-posed problem because there are many shape candidates for a single image. However, humans can easily imagine a 3d shape, so many researchers believe there exists a shape prior learned from past experience [23]. Humans can even use this prior to reconstruct novel objects, and Zhang *et al*. [23] shows that a carefully designed model can learn an implicit representation of this prior and achieve good reconstruction results for both seen and unseen classes.

Inspired by generalized cylinder, We argue that this prior can be explicitly described and applied to shape completion. Generalized cylinder, introduced in [10] as generalized cone, refers to the surface created by moving a cross section along a given smooth axis. Therefore, if an object is a generalized cylinder, we can easily reconstruct its shape

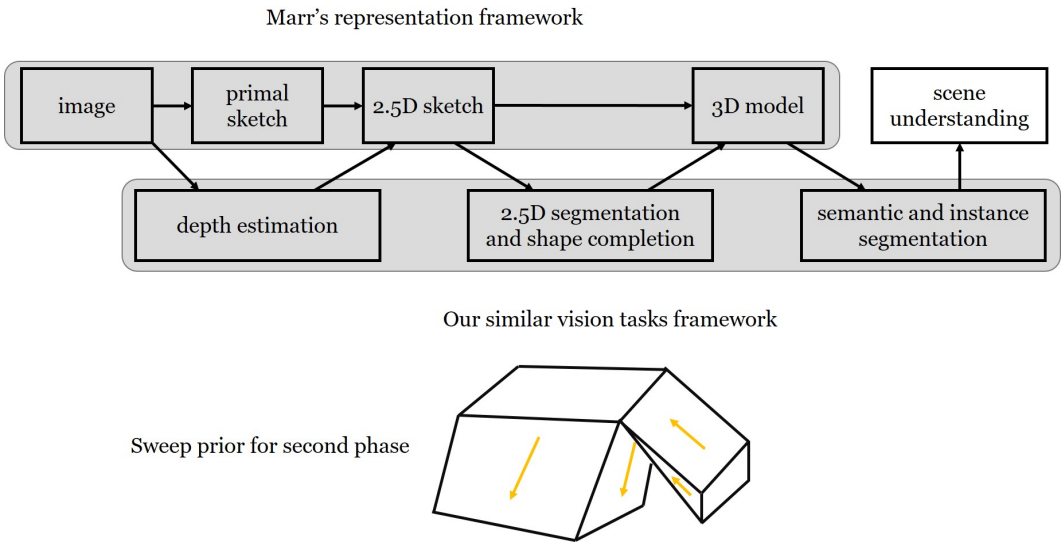

Figure 1. Demonstration of vision tasks framework. Inspired by Marr's theory, we arrange vision tasks across these three representations: image, 2.5D sketch, and 3D model representation. Depth estimation is performed first. Humans can understand novel 3D scenes, so if vision is a process, 3D model extraction must happen before semantic cognition. Accordingly, our second phase is 2.5D segmentation and shape completion of unknown objects. We also find that these two tasks have a tight relationship to each other, and propose sweep prior to tackle them both.

by finding out the cross section and axis from just a fraction of the surface. However, not all objects can be decomposed into generalized cylinder because of its constrains: the shape of cross section must remain constant, and the axis must stay perpendicular to cross section.

To represent more objects, generalized cylinder can be further "generalized" by abandoning these constrains. Consider an inverse process: cutting a compact 3D object into 2D slices. Obviously, no matter what cutting trajectory we choose, we can always get a series of slices, and those two slices near enough can be seen as one sweeping to another. So all objects can be decomposed into parts that are formed by a non-constant cross section moving along a non-perpendicular axis. Though there are numerous ways of decomposition, the choice that has least variation of cross section is regarded as the best. We name this tendency of constructing 3D objects as sweep prior, and show that for any compact 3D objects, inferring the cross section and axis from visible surfaces is possible.

Sweep prior can also help with 2.5D segmentation. The idea is straight forward, if two surfaces can't achieve completion of one shape together, then they must be separated to different object parts. By using this simple prior, we can bridge the gap between 2.5D and 3D, and provide a view-independent representation for the next semantic phase. The vision tasks framework and sweep prior are illustrated in Figure 1.

There are three contributions in our work. First, we introduce a framework similar to Marr's representation framework to unify vision tasks. Second, we show the strong connection between shape completion and 2.5D segmentation, two tasks of our framework's second phase, and propose a prior named as sweep prior. Based on sweep prior, we demonstrate a theory and an algorithm to solve these two tasks together. We also discuss the limitation of our algorithm at last.

2. Related work

Generalized Cylinder. In [10] chapter 3 From Image to Surfaces, Marr defines generalized cylinder as the surface created by moving a cross section alone an axis. He believed that "generalized cones will play an intimate role in the development of vision theory". Though computer vision researchers pay little attention to it recently, generalized cylinder still has some remarkable applications in computer graphics. Chen *et al*. [2] provide user an interactive tool to extract 3D objects by decomposing objects into generalized cylinders. Zhou *et al*. [24] use generalized cylinders to decompose a complex model into simpler parts, and produce fewer parts compared to convex decomposition. In [5], generalized sweep, similar to generalized cylinder, is proposed to re-create CAD model from actual physical parts.

Segmentation of unknown objects. Recently more and more deep learning models have been designed for 3D object classification, detection and segmentation [12, 13, 11, 7]. However due to these methods' supervised nature, it's not suitable to directly apply them to novel objects. For unknown scenes, a large amount of litera-

ture [4, 9, 20, 16, 17, 14] leverage local or global convexity of surface to segment objects into meaningful parts. However, if two objects have surfaces touching each other in convex relation, these two objects won't be segmented apart.

Shape completion of unknown objects. With the development of large CAD model repositories and deep networks, many well-performed methods [18, 19, 8, 25, 1, 6, 3] are proposed to recover shapes of training classes, but tend to have poor generality for novel objects. By using 2.5D sketches and spherical map, Zhang *et al*. [23] achieve state-of-the-art performance for reconstruction of objects within and outside training classes.

3. Theory

All 3D objects can be regarded as the decomposition of parts that are constructed by a cross section sweeping along an axis. However, the possibilities of decomposition are unlimited, same as the possibilities of hiding things behind the visible surface. Therefore, we need some criteria to find the most natural decomposition.

Our first principle is Occam's razor: a valid object must contribute to the visible surfaces. This principle seems self-evident and useless, but we can derive an important conclusion from it: a fraction of an object's visible surfaces must be constructed by cross section's movement. According to our sweep prior, the surfaces can be classified into two kinds, cross section, and surfaces formed by cross section's movement. So if the object's visible surfaces are all cross section, the axis must point into the surface and be occluded. In this case, we would like to think it as 3D objects with small thickness, or 2D objects because of Occam's razor. Obviously, all visible surfaces can be seen as 2D objects, but if a fraction of them can be seen as surfaces of a reasonable 3D object, we prefer the 3D point of view. In other word, If there are no swept surfaces, there are no 3D objects.

The second principle is that the variation of cross section should be small. Take a cube for example. We can take any direction as axis, but if we choose diagonal of the cube, the cross section will change a lot in shape and area. If the axis is the direction of edge, the shape of the cross section will stay constant. However, it's difficult to define the variation of cross section for a general shape. Therefore, we assume another empirical conclusion from this principle: Though cross section itself can be occluded, after sweeping, its next cross section edges must be occluded or seen in visible surface as geometry edges, otherwise the shape and size variation of cross section will be large. We illustrate these two conclusions in Figure 2.

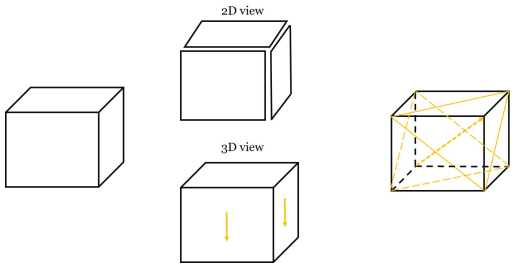

Figure 2. Demonstration of our theory. For an object, we can always see it as composition of 2D objects, but we prefer 3D explanation if possible. If the axis is along diagonal, the cross section edges will appear in surfaces, meanwhile its variation will be large.

4. Algorithm

Leveraging these two conclusions, we propose an efficient algorithm for shape completion and 2.5D segmentation. Our algorithm consists of four parts.

4.1. Geometry Edge Extraction

Based on our theory, there must exist visible swept surfaces for 3D objects, and geometry edges of those surfaces indicate the axis of the shape. In general, geometry edges are defined as discontinuities in surfaces or intersection of two planes. They can further be classified into occluding, occluded, convex, and concave edges. At the surface discontinuities, occluding edges are the points on the foreground, while occluded edges are the corresponding points on the background. When two planes intersect, imagine that we select two points from these two planes then connect them to get a line. The edges are concave edges if the line are visible, otherwise they are convex edges.

For curved surfaces, we assume that they can be constructed as the tensor product of two NURBS curves, or NURBS surfaces, a common representation of free-form surfaces in CAD models. Similar to swept surfaces, NURBS surfaces can be seen as surfaces produced by moving a smooth curve along another smooth curve. By converting NURBS surfaces to regular quadrilateral mesh, we can extract nice geometry edges from the mesh wire-frame. Also, the edges are classified into convex or concave edges using the same criteria.

4.2. Occlusion Inference

After extracting geometry edges, visible surfaces are split to polygons to handle.

When there exist concave or occluded edges on a polygon, it often indicates that part of the polygon is occluded. Inferring the whole polygon from those edges is necessary because it will influence the neighborhood relation of polygons in the next step. We only care about the concave or occluded edges on the outermost boundaries, because those

edges inside the polygon will cause holes that can be filled simply by ignoring them.

First, we find connected components of concave and occluded edges on the boundaries. For one component, we remove it directly and reconnect two edges this component connecting before, or extend and join them. If two edges coincides in line, it means that an edge of the polygon is occluded, so we connect two edges and produce one straight line; If two edges are nearly parallel, at least one edge and two corners are occluded. Though there could exist other possibility, we choose this simplest one and just connect those two edges. A special situation is that the component is one edge. In this case, removing it then connecting will produce the same edge. Though it seems unnecessary, this process changes the property of the edge and allow us to find other neighbor polygon through this edge. Finally, if two edges are not parallel, it suggests that one corner of the polygon is occluded, so we extend two edges to make them intersect. We illustrate occlusion inference in Figure 3.

4.3. Vertex Graph

After getting the real boundaries of polygons, we further build a vertex graph according to the polygons neighborhood relationship.

There are three kinds of graph to represent the relationship: surface graph, edge graph, and vertex graph. Surface graph is widely applied in 2.5D segmentation probably because it's easiest to construct among those three. By assigning similarity according to two surfaces' color, convexity and so on,Weikersdorfer *et al*. [20] uses spectral decomposition, and Richtsfeld *et al*. [15] uses graph cut to figure out if those surfaces belong to the same objects or need to be segmented. However, we argue that these algorithms lack a clear physical meaning, because similarity of two surfaces can't determine whether they are surfaces of same 3D objects.

Vertex graph contains more information than surface graph. We regard two surfaces adjacent if they have one common edge except occluded edges in their real boundaries. Common edges is defined as edges have coincident start and end vertices, so surfaces have coincident edges only are not seen as adjacent surfaces. In this way, we will obtain separated vertex graph sometimes and some vertices may appear more than once, but two objects can be segmented directly, because two surfaces have no common edges must belong to different object parts.

After building vertex graph, we are aware of how many edges one vertex has, or degree of vertex, what surfaces the edge belong to, and the property of edges.

4.4. 3D Inference

Starting with visible surfaces of a scene, we finally obtain vertex graph, an abstract representation suitable for computation and inference.

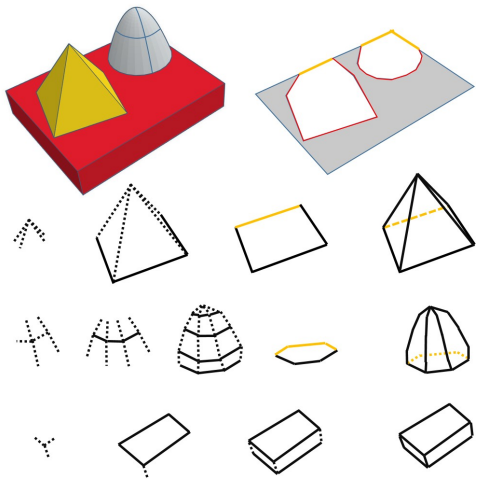

Figure 3. Demonstration of our algorithm. We extract geometry edges first, and perform occlusion inference to find real edges of polygons for vertex graph. After building vertex graph, we start from high degree vertex, then obtain their cross section edges and axis. Finally, objects can be segmented and reconstructed using sweep prior.

As our theory demonstrates, The key for 3D inference is to find the geometry edges of swept surfaces, because those edges indicate the axis of the object part. What's more, the corresponding cross section are often the surfaces adjacent to swept surfaces, and we can reconstruct 3D shape and segment 3D objects accordingly. We know the existence of swept surfaces, but how to find them out?

In vertex graph, a vertex with more than two degree can be seen as intersection of several polygons, and there must exist at least one swept surface among those polygons.

Degree four vertex is generally vertex on curved surfaces, so all the four surfaces are swept surfaces in this case. Additionally, the cross section is occluded, but half of the cross section's edges are the geometry edges on curved surfaces. Therefore, two edges of the degree four vertex are edges of cross section. Another case is quadrangular pyramid, in which all the edges of the degree four vertex are produced by sweeping. Vertex with five degree or higher are similar to the pyramid case.

There are four possibilities for degree three vertex. Take some geometry primitives for example. A cube has two kinds of degree three vertex: vertex with three visible polygons, and vertex with two visible and one occluded polygons. For these two kinds, any two of the three polygons can be swept surfaces. Notice that the occluded polygon may not exist, but we assume its existence if the surface produced by the edges could be hidden behind the visible surfaces. The third kind is vertex of tetrahedron. For a tetrahedron, three polygons of the vertex can be swept surfaces,

but all the other vertices should be the kind of vertex having two visible and one occluded polygons. The last kind is vertex on curved surfaces' occluding edges. This kind of vertex has only two polygons, and both polygons are swept surfaces.

Vertices are not independent to each other. If we choose one possibility of vertex, the vertices around this vertex should be influenced because they have common edges. We start our inference with the vertex having highest degree due to its small possibilities.

All the edges of degree five or higher vertex are produced by sweeping, so edges connecting any two of swept edges are edges of cross section. If cross section edges lie on a visible surface simultaneously, then this surface is the cross section we are seeking for, otherwise the cross section is occluded and need to be inferred from those edges. We can simply connect the start and end vertex of those edges, or fit some common 2D shapes like rectangle, circle or cylinder.

For degree four vertex, we regard it as quadrangular pyramid vertex if there are two occluding edges so the visible surface around the vertex don't cover whole angle, and we deal with it similar as high degree vertex. If it covers, the vertex must lie on curved surfaces, so two of the edges should be edges of cross section, the other two are edges produced by sweeping.

Due to its symmetry, which two are cross section edges can't be determined directly. Accordingly, we break this symmetry by finding the degree four vertex with one degree three neighbor, the vertex lying on occluded edges. In general, occluded edges of curved surfaces are caused by the sweep of cross section's occluding points, so the degree three vertex's remaining edge is the edge of cross section. This edge also belongs to the degree four vertex neighbor. With one edge property known, the property of other three edges can be inferred based on their angle to the known edge. For the same reason, degree four neighbor of the known degree four can be inferred. This process will continue until it hits another degree three vertex. What's more, the edges touching those cross section edges are all swept edges, along which we can move the cross section to obtain next group of cross section edges. Finally, if the cross section edges arrives at a visible surface, we can get the whole cross section directly, otherwise we need to infer the whole cross section as before.

For remaining degree three vertex, we select two edges of the vertex as cross section edges. Though it's possible that all the three edges are swept edges, tetrahedron for example, we notice this case can be handled passively, because there must exist other kind of degree three vertex accompanying it. We can get the common polygon of the selected two edges, and prefer to choose the polygon with the largest number of edges as cross section. the inference process is illustrated in Figure 3.

After getting cross section and the swept edges, we can reconstruct 3D shape of relevant visible surfaces and segment them apart from other surfaces at the same time. Compared to segmenting surfaces by convexity, our method can provide a more reasonable explanation: they belong to different 3D objects.

4.5. Limitation

Surfaces are an ideal representation of the 2.5D sketch. In our algorithm, we assume that we already know curved surfaces and their NURBS parameters, but it's actually difficult and time-consuming to fit arbitrary NURBS surfaces on raw noisy data like depth or point clouds.

There are also some failure cases in other steps. If severe occlusion occurs, for instance, a cylinder's body is occluded thoroughly, then the cylinder's head and tail will be regarded as two 3D objects because the two parts have no adjacent surfaces. What's more, the composition of objects in the scene can be too complex for our vertex graph to handle. For example, If a 2D object like a card put on the surface of 3D objects like a cube, and part of the card stands out, our algorithm can't distinguish the card from the surface so the inference fails. Another case is concave objects like a cup or bowl. A natural way to understand these is subtracting an 3D parts from a compact objects, which is impossible in our current algorithm however.

Nevertheless, we argue that those situations don't conflict with the sweep prior because of a simple reason: we can imagine these situations by composition of typical swept objects. Accordingly, these limitations don't conflict with sweep prior, and we can rely on the theory still to develop a better algorithm in the future.

5. Conclusion

In this work, we have presented a vision tasks framework similar to Marr's representation framework, and handled the second phase, from 2.5D to 3D, leveraging a simple prior called sweep prior. Compared to convexity based or learning based methods, our approach provides a meaningful and unified explanation for segmenting and reconstructing novel objects. Though our algorithm has many limitations, our theory's simplicity and the unification of two tasks make us believe that sweep prior can be further applied to more challenging 3D scenes, and will play a key role in 3D scene understanding.

References

[1] A. Arsalan Soltani, H. Huang, J. Wu, T. D. Kulkarni, and J. B. Tenenbaum. Synthesizing 3d shapes via modeling multi-view depth maps and silhouettes with deep generative networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1511–1519, 2017. 3

[2] T. Chen, Z. Zhu, A. Shamir, S.-M. Hu, and D. Cohen-Or. 3-sweep: Extracting editable objects from a single photo. *ACM Transactions on Graphics (TOG)*, 32(6):195, 2013. 2

[3] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, pages 628–644. Springer, 2016. 3

[4] S. Christoph Stein, M. Schoeler, J. Papon, and F. Worgotter. Object partitioning using local convexity. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 304–311, 2014. 3

[5] M. Goyal, S. Murugappan, C. Piya, W. Benjamin, Y. Fang, M. Liu, and K. Ramani. Towards locally and globally shape-aware reverse 3d modeling. *Computer-Aided Design*, 44(6):537–553, 2012. 2

[6] C. Häne, S. Tulsiani, and J. Malik. Hierarchical surface prediction for 3d object reconstruction. In *2017 International Conference on 3D Vision (3DV)*, pages 412–420. IEEE, 2017. 3

[7] M. Jiang, Y. Wu, T. Zhao, Z. Zhao, and C. Lu. Pointsift: A sift-like network module for 3d point cloud semantic segmentation. *arXiv preprint arXiv:1807.00652*, 2018. 2

[8] A. Kar, S. Tulsiani, J. Carreira, and J. Malik. Category-specific object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1966–1974, 2015. 3

[9] A. Karpathy, S. Miller, and L. Fei-Fei. Object discovery in 3d scenes via shape analysis. In *2013 IEEE International Conference on Robotics and Automation*, pages 2088–2095. IEEE, 2013. 3

[10] D. Marr. Vision: A computational investigation into the human representation and processing of visual information, henry holt and co. *Inc., New York, NY*, 2(4.2), 1982. 1, 2

[11] C. R. Qi, O. Litany, K. He, and L. J. Guibas. Deep hough voting for 3d object detection in point clouds. *arXiv preprint arXiv:1904.09664*, 2019. 2

[12] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017. 2

[13] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*, pages 5099–5108, 2017. 2

[14] A. Richtsfeld, T. Mörwald, J. Prankl, M. Zillich, and M. Vincze. Segmentation of unknown objects in indoor environments. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4791–4796. IEEE, 2012. 3

[15] A. Richtsfeld, T. Mörwald, J. Prankl, M. Zillich, and M. Vincze. Learning of perceptual grouping for object segmentation on rgb-d data. *Journal of visual communication and image representation*, 25(1):64–73, 2014. 4

[16] S. C. Stein, F. Wörgötter, M. Schoeler, J. Papon, and T. Kulvicius. Convexity based object partitioning for robot applications. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3213–3220. IEEE, 2014. 3

[17] A. Ückermann, R. Haschke, and H. Ritter. Realtime 3d segmentation for human-robot interaction. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2136–2143. IEEE, 2013. 3

[18] P.-S. Wang, Y. Liu, Y.-X. Guo, C.-Y. Sun, and X. Tong. O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *ACM Transactions on Graphics (TOG)*, 36(4):72, 2017. 3

[19] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon. Dynamic graph cnn for learning on point clouds. *arXiv preprint arXiv:1801.07829*, 2018. 3

[20] D. Weikersdorfer, D. Gossow, and M. Beetz. Depth-adaptive superpixels. In *Proceedings of the 21st international conference on pattern recognition (ICPR2012)*, pages 2087–2090. IEEE, 2012. 3, 4

[21] J. Wu, Y. Wang, T. Xue, X. Sun, B. Freeman, and J. Tenenbaum. Marrnet: 3d shape reconstruction via 2.5 d sketches. In *Advances in neural information processing systems*, pages 540–550, 2017. 1

[22] A. R. Zamir, A. Sax, W. Shen, L. J. Guibas, J. Malik, and S. Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3712–3722, 2018. 1

[23] X. Zhang, Z. Zhang, C. Zhang, J. Tenenbaum, B. Freeman, and J. Wu. Learning to reconstruct shapes from unseen classes. In *Advances in Neural Information Processing Systems*, pages 2257–2268, 2018. 1, 3

[24] Y. Zhou, K. Yin, H. Huang, H. Zhang, M. Gong, and D. Cohen-Or. Generalized cylinder decomposition. *ACM Trans. Graph.*, 34(6):171–1, 2015. 2

[25] J.-Y. Zhu, Z. Zhang, C. Zhang, J. Wu, A. Torralba, J. Tenenbaum, and B. Freeman. Visual object networks: image generation with disentangled 3d representations. In *Advances in Neural Information Processing Systems*, pages 118–129, 2018. 3