Clustering algorithms used in 3D scene segmentation

Isma Hadji* and Daniel Nabelek**
Department of Electrical and Computer Engineering
University of Missouri
19 Eng. Building West, Columbia, MO, 65211
Email: *ih9p5@mail.missouri.edu, **dpn6f4@mail.missouri.edu

***Abstract*—In this paper, we implement and compare three different clustering algorithms for the purpose of 3D image segmentation. Specifically, the K-means, Mean Shift, and Hierarchical methods are studied, and their performance is compared using cluster validity methods. Performance was analyzed in two ways, first by comparing independent results from each, and second, by comparing results where Hierarchical clustering is used as a cluster reduction method, following K-means and Mean Shift. Experiments show that each method is robust and can produce a clustering of the 3D data that, when compared to a ground truth using cluster validation, can consistently produce a Rand statistic greater than 0.7.**

I. Introduction

In robotic vision it is of paramount importance to understand scenes and what robots may be looking at. For this, algorithms are required to recognize objects and determine their location. The first step towards achieving this task is to segment the scenes in a way that every segment represents a different object in the scene. Depending on the scene this task might be very difficult especially if we are dealing with 2D images. Although, the appearance of range sensors –e.g. Kinect; makes the task easier, thanks to the availability of 3D information, it is still a big challenge to achieve perfect segmentation for more complex scenes.

A very natural way to address the problem of segmentation is clustering points based on their euclidean distance. For this reason, in this work we are tackling the problem of 3D scene segmentation using several clustering algorithms. The objective of this work is twofold. The first purpose is to evaluate how one clustering algorithm may be more suitable than another, as a result of differences with the logic used to find the clusters. The second and most important goal is to demonstrate how different clustering algorithms can be used together to achieve better segmentation results. We are proposing in this work a coarse to fine clustering scheme. First by clustering the points in the scene in such a way that we end up "over-segmenting" the scene. Then using the resulting segments and clustering those in order to find the final clusters, following a merging scheme.

The rest of this paper is organized as follows. Section II briefly presents some of the clustering algorithms that have been used in the computer vision field for the task of scene segmentation. In section III we describe each one of the clustering algorithms evaluated in this paper. Section IV thoroughly describes the experiments made, the dataset used and presents the obtained results. Finally, we summarize with a conclusion.

II. Background and Related Work

Although 3D scene segmentation can be addressed using different approaches, like region growing based on distance, curvatures or colors; clustering based segmentation remains one of the most widely used techniques. Usually, in order to segment 3D scenes, approaches inspired from 2D are used. These methods often involve a pre-processing step in order to find keypoints in the scene and clustering is applied to the keypoints to find the final segments. Among the most prominent works falling in this category we find [1] that clusters SIFT keypoints [2] extracted from RGB and Depth images using K-means. Similarly, [3] follows the same framework of clustering keypoints, but using an incremental clustering in which Hierarchical clustering is used to find and merge clusters.

Hierarchical clustering is well established as an image segmentation method. The bottom up approach to hierarchical analysis is described in [4], where the bottom level of the hierarchy corresponds to the input image and the top level describes the result of the hierarchy. Clustering at each level is based on the criterion that two adjacent clusters are merged if an only if they are mutually nearest neighbors. A common approach as presented in [5], is the use of over-segmentation into small segments called superpixels followed by a hierarchical merging algorithm with some merging criterion. This was motivated by the study of electron microscopy of neural circuits, which results in huge volumes of image data. Additionally, segmentation of 3D LIDAR data using hierarchical clustering is described in [6] for the purpose of estimating the number of tree stems present in LIDAR forest data. Many other applications of hierarchical clustering algorithms can be found ranging from social sciences to archaeology [7].

Hierarchical clustering can be divided into two main categories of algorithms, agglomerative and divisive [7]. Agglomerative clustering can be described as a bottom-up approach, initialized so that each of the N clusters contains only a single element X. At each step, two clusters are merged based off some criterion producing N – 1 clusters. In this work we explore only the agglomerative approach.Alternatively, divisive hierarchical clustering follows the inverse path [7] and can be described as a top-down approach. In this case only a single cluster exists (N = 1) containing all the elements X. At each step, one of the clusters is split based off some criterion. This results in N ¿ 1 clusters containing only a single element. Divisive clustering is not commonly done in practice due to the fact that it is computationally expensive [8].

Another widely used technique for scene segmentation is Mean Shift algorithm. The most famous work applying Mean Shift to the task of image segmentation has been introduced in [9] for 2D images where Mean Shift has been proven to achieve excellent results for image segmentation. Ever since many improvements have been introduced to the original Mean Shift algorithm to improve its efficiency. Some examples include [10]that proposes adaptive mean shift in which the neighborhood of each seed point is used to estimate the bandwidth required by the Mean Shift algorithm. Similarly, [11] introduces the use of surface curvatures to make the original Mean Shift algorithm more dedicated to 3D applications.

In this work, we chose to use 3D cloud of points directly for scene segmentation without introducing any pre-processing step. We propose to do a comparative study of several clustering algorithms in terms of how suitable they can be for our specific application. This work is most similar to [12] that uses 3 different clustering algorithms. However, while [12] focuses on the importance of 3D information, in this work we focus on the role of the clustering algorithm itself.

III. Clustering Techniques

In this work we are comparing different clustering algorithms for the task of 3D scene segmentation. Although several clustering algorithms can be used for this purpose, we have narrowed down the suitable algorithms for our task to three main clustering algorithms. Namely, we will study the clustering capability of 1) K-means, , 3) Mean Shift, and finally the Hierarchical clustering algorithm. In addition to that, we propose in this work to use the two first algorithms as an initialization step where we allow a relatively large number of clusters to be detected and then use the resulting clusters as input to the Hierarchical clustering algorithm for cluster merging and generation of the final segments. This is what we will refer to in this work as a coarse to fine segmentation scheme.

A. K-means

K-means is probably one of the most widely used clustering algorithms, due to its simplicity and speed. It is therefore the first clustering algorithms that we have tested in this work. The main idea behind K-means is the minimization of the distance between points that are potentially in the same cluster. This is done by minimizing the objective function shown in equation 1. The fact that K-means is based on minimizing distance makes it a very attractive choice for our application, given that we want points that are close to one another in space to fall within the same clusters. However, one major disadvantage in this case is the random initialization of K-means that might throw off the clusters from the very beginning. In addition to that, the number of objects present in the scene should be known before hand to get the right number of segments with K-means. However, K-means remains a good choice in our coarse to fine clustering scheme described earlier in section III.

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} ||x_n - \mu k|| \tag{1}$$

B. Mean Shift

The mean shift algorithm approaches the clustering problem in a somewhat different approach as opposed to the algorithms described in the previous subsection. In fact, mean shift considers the input as a probability density function and its goal is to find the modes of this function which will represent the cluster centers. This is done by applying the gradient ascent method to the kernel density estimate given the input points. The density estimation is given by equation 2 where N is the total number of points and h is the bandwidth or the size of the window used to estimate the density using the kernel function K. The gradient is given by equation 3 which when set equal to 0, gives us the mean shift vector, described by equation 4.

$$f(x) = \frac{1}{Nh^d} \sum_{i=1}^{N} K(\frac{x - x_i}{h}) \tag{2}$$

$$\nabla f(x) = \frac{1}{Nh^d} \sum_{i=1}^{N} K'(\frac{x - x_i}{h}) \tag{3}$$

$$mx^{\rightarrow} = \frac{\sum_{i=1}^{N} K'(\frac{x-x_i}{h}) x_i^{\rightarrow}}{\sum_{i=1}^{N} K(\frac{(x-x_i)}{h})} \tag{4}$$

So in sum Mean Shift is built around 2 main steps. First computing the mean shift vector of the data within a window specified by the bandwidth, then shifting the window to the mean and repeating until the moves are not significant. This is done for every point and the points that fall within the same basin at the end form a cluster.

The fact that we don not have to provide an estimate of the number of clusters that we expect in the scene using mean shift makes it a very attractive choice. The only value that we need to determine is the bandwidth size. However, as previously mentioned in the background section, the bandwidth can be estimated from each points closest neighbor as is proposed in adaptive mean shift [10]. In this work we have considered using Adaptive mean shift as well in order to judge on the effect of the bandwidth and the neighborhood size.

C. Hierarchical Clustering

In this study we examined the use of agglomerative hierarchical clustering in two ways. First, we used it as a method for producing an independent clustering of 3D data. Secondly, we used it as a method for reducing the numbers of clusters produced using the other clustering algorithms previously described. The agglomerative algorithm can be described as follows [7], from which it is clear that we start with many clusters containing only a single element and end with one cluster containing all elements.

- Create initial clustering $R_0 = \{C_i = \{x_i\}, i = 1, ..., N\}$. and $t = 0$
- Repeat until only one cluster exists
  - $t = t + 1$
  - Check all possible pairs of clusters (C_r, C_s) and find one pair (C_i, C_j) such that $g(C_i, C_j) = \Big\{min(r, s) \quad \vee max(r, s)\}$ depending on whether g is a dissimilarity or similarity function respectively.
  - Merge clusters C_i and C_j such that $C_q = C_i \cup C_j$, producing a new cluster $R_t = (R_{t-1}) - \{C_i, C_j\} \cup \{C_q\}$

When hierarchical clustering was used as a standalone clustering method we were able to use the MATLAB function *linkage* to compute the multilevel cluster hierarchy. We can then use the function *cluster* to compute the cluster labels at a certain level in the hierarchy (set number of clusters). Ward's algorithm was used as the criterion for cluster merging when using the linkage function. This algorithm is a minimum variance algorithm where the distance between two clusters C_i and C_j is given by equation 5,

$$d'_{ij} = \frac{n_i n_j}{n_i + n_j} d_{ij} \tag{5}$$

In our work d_{ij} gives the euclidean distance between data points. Note that this only includes the x, y, z location of each pixel and not their color values. The above equation is then minimized when determining the two clusters to merge. The entire hierarchy is computed all the way to the final clustering where only one cluster remains. The hierarchy is then explored, from the top down, until level is reached where the specified number of clusters exists. The cluster memberships at this level are then used as our output clustering result.

After looking at hierarchical clustering as an independent clustering method, we also looked at it as a way to reduce the number of clusters produced by other clustering algorithms, specifically K-means and Mean-shift. Unlike the independent method, the algorithm had to be coded without the use of the linkage function due to the fact that the hierarchical method is no longer starting at the bottom of the hierarchy. In this case, we have an initial clustering where the number of clusters is less than the total data points and clusters may contain more than one data point.Following this initialization, clusters are merged again using Ward's criterion where the two clusters that are merged minimize **??**. The merged clusters are then represented as the centroid of the resulting cluster. This process continues until the desired number of clusters, less than the starting number, is obtained.

IV. Experimental Results and Discussion

We have divided our experiments into two main steps. In the first step, we perform a comparative evaluation of the 3 clustering algorithms considered. In the second step we use K-means and Mean Shift in coarse steps to produce over-segmented scenes that we fine tune using Hierarchical clustering. We have performed our experiments on 4 different scenes with a varying level of clutter and a variable number of objects. Figure 1 depicts the used scenes. We have performed our experiments with the goal of putting the different objects in different segments. It is worth noting here, that for all clustering algorithms we relied solely on the 3D coordinates of the points as our base features to segment the scenes.

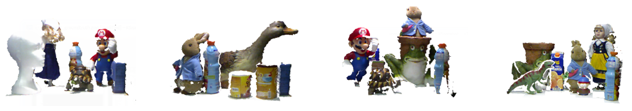


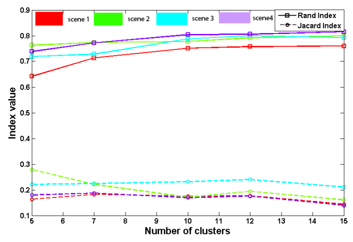
Figure 1. The dataset used for the experiments with 4 different scenes

A. *Comparative Evaluation of Different Clustering Algorithms*

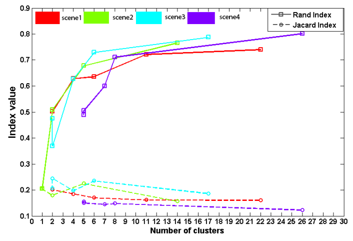
Our goal here is to first compare the effectiveness of the three different clustering algorithms independently. For doing that, we allow the clustering results of the different algorithms to vary based on the heuristics they rely on. For K-means we used cluster populations of K={5,7,10,12,15} while for Mean Shift we vary the bandwidth from 0.05 to 0.25 in increments of 0.05. Also, for the case of Hierarchical clustering we decided to cut the results of the agglomerative clustering to have the same number of clusters as was done for K-means.

In order to evaluate the algorithms both individually and with respect to one another, we manually annotate the scenes with the ground truth such that every object in the scene belongs to a different cluster. For our evaluation, we chose to use both the Rand as well as the Jaccard index. Figure 2 illustrates the variations of the indices with respect to the cluster numbers for the different scenes. As we can see from figure 2 the Rand index is relatively high for all experiments and especially for K-means and Hierarchical clustering, going over 0.7 in most of the experiments. This demonstrates the consistency of those 2 methods. Interestingly enough, these same results highlight the difference between the 2 indices, where the Jaccard index is drastically smaller as opposed to the Rand index. This tells us that although the clustering is consistent overall, a lot of points remain in different clusters, which explains the drop in the Jaccard index. This observation is valid for all algorithms.

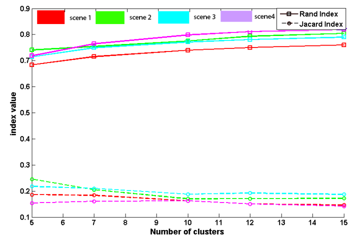
On the other hand, comparing the 3 algorithms we notice that the clustering capability of both K-means and Hierarchical clustering are pretty steady and comparable, which is probably explained by the fact that a similar number of clusters is used for both. Mean shift on the other hand struggles more as the bandwidth is small. This is mainly due to the fact that Mean Shift fails to find different clusters as the bandwidth increases and ends up putting all points in one or 2 clusters at most. However, as the bandwidth gets smaller, and the number of detected clusters bigger, Mean Shift catches up with the other 2 techniques. To overcome the susceptibility of Mean Shift



(a)



(b)



(c)

Figure 2. Rand and Jaccard index variations with respect to the number of clusters for the 4 evaluated scenes using (a) K-means, (b) Mean Shift and (c) Hierarchical Clustering

to the bandwidth value, we have considered using Adaptive Mean Shift. For doing so, we modified the original Mean Shift code and used the furthest away point among the K nearest neighbors. Given that the size of each point cloud is around 50000 to 70000 each we have set the K nearest neighbor value to 5000, which detected 8, 7, 9 and 9 clusters for the 4 respective scenes.

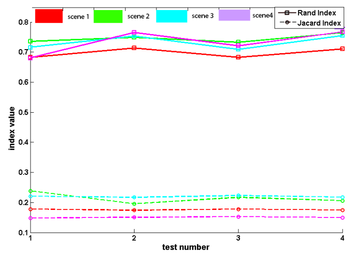
Overall, we notice that detecting a higher number of clusters increases the validity indices. Figure 3 illustrates the best segmentation results according to the highest validity index. Although, higher number of clusters improve the quantitative results, the qualitative results still need to be improved. This motivates our next experiment where we attempt to improve the quantitative results using a coarse to fine approach.

B. Coarse to fine clustering using a Combination of Clustering Algorithms

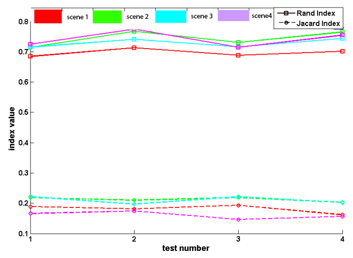
Using the methods presented in the previous section, clustering results for both the K-means and Mean Shift algorithms were generated. The number of clusters for each was desired to be relatively large, and two clustering results for each algorithm were used. Clustering results for the K-means algorithm, which requires the number of clusters decided beforehand, contained 30 and 50 clusters for each scene. The number of clusters for the Mean Shift results was based of the bandwidth. These clusters were then used as the initialization for the hierarchical algorithm. Table4 gives the start/end population of each test before and after the hierarchical clustering. At each step, this algorithm compared every pair of clusters using Ward's criterion. The pair of clusters that minimized **??** were then merged until the desired number of clusters was obtained. Again, the cluster results of each test were compared using the Rand and Jaccard statistics. Figure5 shows these statistics plotted for each test number.

K-means/Hierarchical						Mean Shift/Hierarchical				
Scene	Test 1	Test 2	Test 3	Test 4		Scene	Test 1	Test 2	Test 3	Test 4
1	30/5	30/7	50/5	50/7		1	46/5	46/7	34/5	34/7
2	30/5	30/7	50/5	50/7		2	39/5	39/7	22/5	22/7
3	30/5	30/7	50/5	50/7		3	37/5	37/7	25/5	25/7
4	30/5	30/7	50/5	50/7		4	62/5	62/7	39/5	39/7

Figure 4. Starting/ Ending Cluster Population



(a)



(b)

Figure 5. Rand and Jacard index variations with respect to the number of clusters for the 4 evaluated scenes using Hierarchical clustering applied to (a) K-means, (b) Mean Shift

This data shows that we do not get any clear performance gain by adding the hierarchical step to our clustering. Possible reasoning for these findings is that while hierarchical clustering independently goes through over 50,000 cluster reductions on this dataset, adding a hierarchical step with no more than 50 cluster reductions may not produce much of a change. Perhaps differences in the clustering results would become more apparent if the hierarchical clustering step operated over relatively more cluter reductions.

V. Conclusion

Results show that each of the evaluated clutering methods is robust for the purpose of segmenting the 3D data used. The Rand and Jaccared indices were a useful measurement in this evaluation. Using these, we were able to demonstrate that

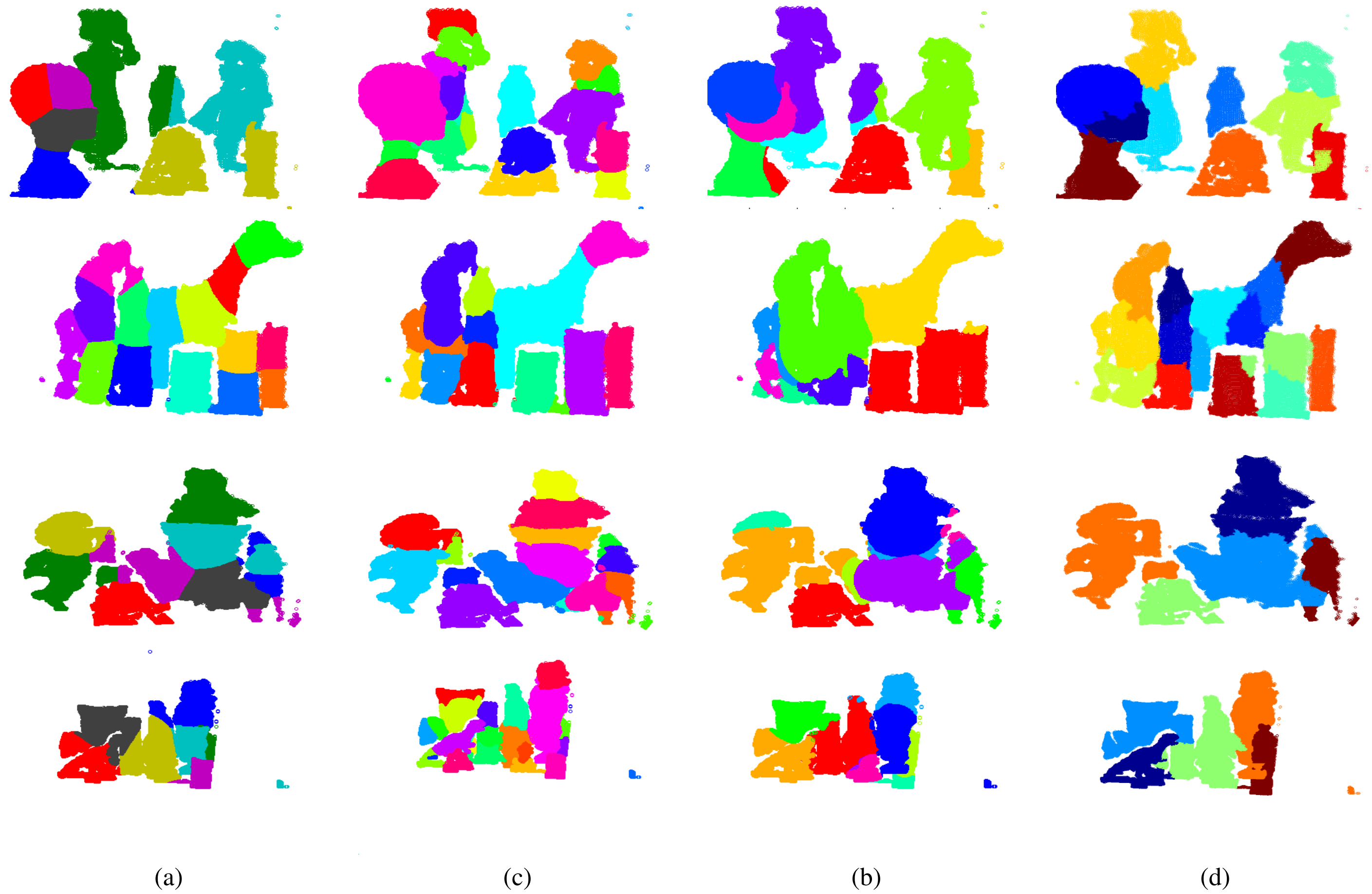


Figure 3. Segmentation results using different clustering algorithms. (a) K-means,(b) Mean Shift, (c) Adaptive Mean Shift, (d) Hierarchical Clustering. Each row represents a different scene

each algorithm operates fairly consistently on each of the four scenes segmented. However, we were not able to demonstrate an improvement in performance when a hierarchical clustering step is used to fine tune the results. Further experiments should be done to better understand how hierarchical clustering can fit with the other two methods to produce a better clustering scheme. As the Rand and Jaccard indices suggest, it is possible that this coarse to fine hierarchical approach for fine tuning K-means and Mean Shift, could indeed work. Yet for optimum performance, the starting number of clusters for the hierarchical step should be larger so that more cluster reductions are performed hierarchically.

References

[1] N. Silberman and R. Fergus, "Indoor scene segmentation using a structured light sensor," in *ICCV Workshops*. IEEE, pp. 601–608.

[2] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[3] A. Angeli and A. Davison, "Live feature clustering in video using appearance and 3d geometry," in *Proc. BMVC*, 2010, pp. 41.1–11, doi:10.5244/C.24.41.

[4] X. SHEN, M. SPANN, and P. NACKEN, "Segmentation of 2d and 3d images through a hierarchical clustering based on region modelling," *Pattern Recognition*, vol. 31, no. 9, pp. 1295 – 1309, 1998.

[5] J. Nunez-Iglesias, R. Kennedy, T. Parag, J. Shi, and D. B. Chklovskii, "Machine learning of hierarchical clustering to segment 2d and 3d images," *PLoS ONE*, vol. 8, 08 2013.

[6] J. Reitberger, C. SchnÃ¶rr, P. Krzystek, and U. Stilla, "3d segmentation of single trees exploiting full waveform lidar data," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 64, no. 6, pp. 561 – 574, 2009.

[7] S. Theodoridis and K. Koutroumbas, *Pattern Recognition, Fourth Edition*, 4th ed. Academic Press, 2008.

[8] A. Zare, *ECE 8735 Notes Lecture 20, Hierarchical Clustering*. University of Missouri, 2013.

[9] D. Comaniciu, P. Meer, and S. Member, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 603–619, 2002.

[10] D. Comaniciu, V. Ramesh, and P. Meer, "The variable bandwidth mean shift and data-driven scale selection," in *in Proc. 8th Intl. Conf. on Computer Vision*, 2001, pp. 438–445.

[11] X. Zhang, G. Li, Y. Xiong, and F. He, "3d mesh segmentation using mean-shifted curvature," in *Proceedings of the 5th International Conference on Advances in Geometric Modeling and Processing*, ser. GMP'08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 465–474. [Online]. Available: http://dl.acm.org/citation.cfm?id=1792279.1792316

[12] C. D. Mutto, P. Zanuttigh, G. M. Cortelazzo, and S. Mattoccia, "Scene segmentation assisted by stereo vision," in *Proceedings of the 2011 International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission*, ser. 3DIMPVT '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 57–64. [Online]. Available: http://dx.doi.org/10.1109/3DIMPVT.2011.16