Robust Segmentation of cluttered scenes using RGB-Z images

Navneet Kapur and Subodh Iyengar

November 19, 2011

1 Abstract

Image segmentation is a fundamental preprocessing step to other vision tasks such as object recognition. Our project focusses on using depth information from a Kinect depth sensor as an additional feature to aid in segmenting an image. In this report we describe the work we have done so far in the project.

2 Introduction

With the availability of commercial depth sensors such as the Kinect, there is a body of work using the depth information provided by the Kinect to segment images of scenes [1][6]. In this project we try to segment cluttered scenes by augmenting color information in images with depth cues from the Kinect data. In this milestone, our major focus was to study the literature in the RGB-Z domain, implement code to replicate the method in [4], and try different modifications in order to make the method perform better.

3 Related Work

In previous work [6] a top down approach to segmentation is used, by using a combination of a Canny detector, a Delawney Triangulations and modified Normalized Cut [5]. Other work [1] focusses on segmenting planes in real time using the kinect depth sensor. The method presented here is a bottom-up approach to constructing meaningful segments of pixels.

4 Approach

We obtained our Kinect dataset from the Bekeley 3DO project [2]. These consist of a set of images of both RGB as well as depth maps obtained from the Kinect. The 3DO project provides both raw depth map images as well as depth images that have been calibrated so as to map each pixel of the depth image to each pixel in the RGB image. The raw depth image is also smoothed to reduce the jitter that is observed as described in [2]. We thus use the smoothed version of the depth images to obtain depth values for the pixel locations. An example of the depth image can be seen in figure 5. While choosing images from the data set we choose ones that have a depth variance as well as occlusion effects.

Our main task for this milestone was to replicate the method in [4], and produce similar behaviour given different scenes. As the first step, like in [4], we create 200 super-pixels using the N-cut method [5] using publicly available code. This is done using only the color information in the original image. In the second step we incorporate depth information for segmentation.

To incorporate depth information, we consider the super-pixels obtained from the first step as a graph instead of the pixels themselves. Each super-pixel s_i is a node in the graph and there is an edge ϵ_{ij} connecting any pair of adjacent super-pixels s_i and s_j. As postulated in [4], each edge represented as such, has an associated weight, which for a pair of super-pixels s_i and s_j is given by:

$$w_{ij} = \frac{1}{\epsilon_{ij}.|(c_i - c_j)^T n_i|.|(c_i - c_j)^T n_j|}$$

Where c_i, $c_j \in \mathrm{R}^3$, are the centroids of the respective super-pixels , $n_i, n_j \in \mathrm{R}^3$, are the normals of the respective planes in which all points of the super-pixel lie, and e_{ij} represents the error of the points fitting the plane in the combined cluster.

To obtain c_i, we first obtain the x, y, z coordinates of the image point in the world coordinates. c_i is obtained as the centroid of all the points in the super-pixel. To obtain n_i, we formulate the problem as a linear equation $[x, y, z, 1; ....] * [a, b, c, d]^T = 0$ to find the parameters of the plane $ax + by + cz + d = 0$ representing the least squares fit of the points in the cluster. The solution to this is the last column of V in the SVD of $[x, y, z, 1; ....]$. This represents the eigenvector with the least eigenvalue.

Instead of using a priority queue as in [4], we use a matrix of size (initial number of clusters x initial number of clusters) to store the weights between super-pixels. We then perform an iterative procedure to merge super-pixels greedily. We cache all the normals and the centers that are computed for efficiency.

To implement greedy merging, in each iteration, we choose two super-pixels which have maximum weight and merge them. The process is simply a lookup for the maximum element in the matrix that stores the weight values. To merge the 2 super-pixels we assign the super-pixel number of the super-pixel with the lower number to all the elements in both the super-pixels. We then update the n_i and c_i values for the merged super pixel and also update the weights for only those entries which change in the weight matrix, i.e. for all the adjacent nodes of the merged super-pixel and itself.

5 Experiments

We perform 2 types of evaluations on our implementation as well as the modifications we made to the method. One measure is qualitative, by looking at the clusters and commenting on the clustering errors. Another way to measure clustering performance is quantitative. For a quantitative measure, we implemented the entropy measure proposed in [4].

The entropy measure is defined as:

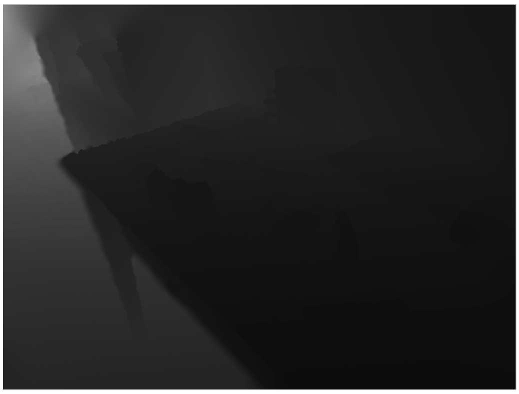

Figure 1: Smoothed Depth image

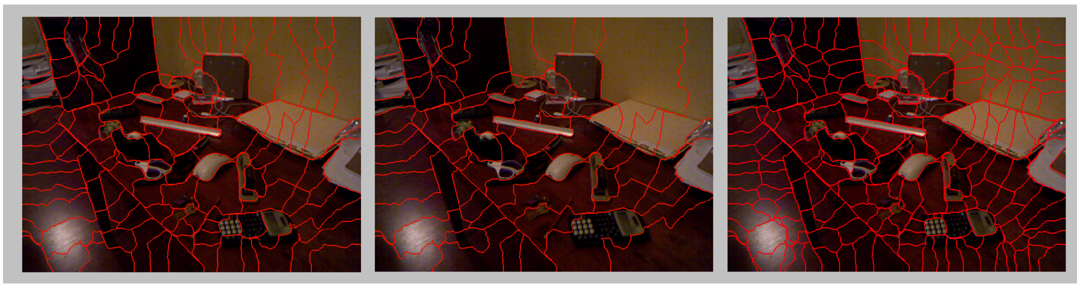

Figure 2: Using metric 1. Left image, 150 clusters (70 iters). Center: 50 clusters (170 iters), Right: 220 clusters (original after superpixelation)

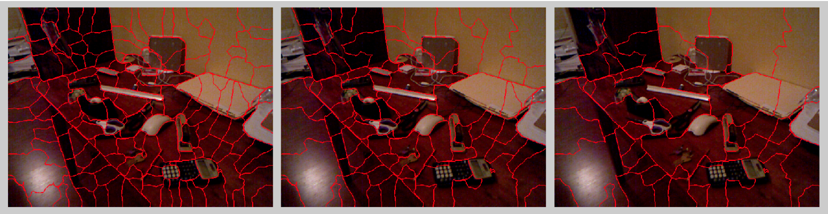

Figure 3: Using metric 2. Left image, 160 clusters. Center: 100 clusters, Right: 50 clusters

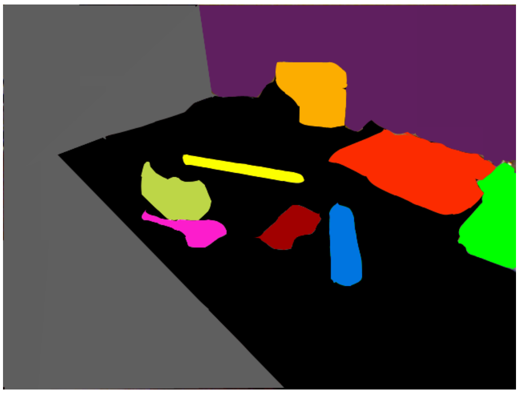

Figure 4: Ground Truth Image

$$I = -\frac{1}{N} * \sum_{i \in S_{gt}} \sum_{j \in L_i} P_{ij} ln P_{ij}$$

Where P_{ij} is the percentage of Label j in segment i. S_{gt} is all the ground truth segments. We manually labeled a ground truth image as shown in figure 5. Th initial entropy values that we got from super-cluster segmentation and merging were clustered around values of -5 (which was a marginal decrease from the entropy of the super-clustering that use as the input for our greedy-merge algorithm).

We also experimented with different weight metrics for the greedy merging. One problem with the previous weight metric is that it might allow for merging of perpendicular surfaces. Since we are merging between segments that are adjacent in each step, we do not need to keep track of weights beyond a radius of 1 from a node. This we can ignore the c_i terms, since for adjacent nodes they will be similar. Thus we define this new metric as:

$$w_{ij} = \frac{1}{e_{ij} * \|n_i - n_j\|}$$

We call this **weight metric 2** and the previous one as **weight metric 1**. The output we obtain from this metric is shown in figure 3.

Results from running the method using metric 1 and metric 2 are shown in figure 2 and figure 3 respectively. We observe the results obtained by merging the segments together. We observe that as suspected, the weight metric 1 merges the back wall with the laptop on the table in figure 2, however with our modified metric 2, for the same number of clusters, the perpendicular planes do not get merged. Otherwise the resulting clusters from the 2 metrics are similar and do not show much difference.

6 Future Work

As work for the final submission we would like to try different ideas to incorporate depth information such as describing each cluster with a feature vector so as to make it easier to apply machine learning to cluster merging as in [3], instead of using heuristic metrics.

References

[1] Dirk Holz, Stefan Holzer, Radu Bogdan Rusu, and Sven Behnke. Real-time plane segmentation using rgb-d cameras. In *RoboCup Symposium*, 2011 2011.

[2] Allison Janoch, Sergey Karayev, Yangqing Jia, Jonathan T. Barron, Mario Fritz, Kate Saenko, and Trevor Darrell. A category-level 3-d object dataset: Putting the kinect to work. In *ICCV Workshop on Consumer Depth Cameras for Computer Vision*, to appear 2011.

[3] X. Ren and J. Malik. Learning a classification model for segmentation. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 10 –17 vol.1, oct. 2003.

[4] Jiahui Shi. Rgb-z segmentation of objects in a cluttered scene using a kinect sensor.

[5] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. In *IEEE Transactions on Pattern Analysis and Machine Intelligence.*

[6] Camillo Taylor and Anthony Cowley. Segmentation and analysis of rgb-d data. Technical report, University of Pennsylvania, 2011.