Learning Planar Geometric Scene Context Using Stereo Vision

Paul G. Baumstarck, Bryan D. Brudevold, and Paul D. Reynolds
{pbaumstarck,bryanb,paulr2}@stanford.edu
CS229 Final Project Report
December 15, 2006

Abstract—**A reliable method for detecting planar regions in a video/stereo scene would be of great use to the field of computer vision. Solutions to this problem are applicable to object recognition, scene identification, and robot-related applications. In this paper we present a plane-finding algorithm that uses data from a binocular stereo camera system to produce labeled output images showing the major planes in a video scene. The algorithm is based on the three-dimensional Hough Transform but also presents many useful approaches and heuristics applicable to general plane-finding.**

I. Introduction

The goal of our project was to use image and depth data from a stereo camera system to locate the major planes in an image. A successful algorithm would find immediate application in areas such as scene identification, object recognition, and robot-environment interaction. After trying several approaches built around unsupervised learning algorithms, we converged on an algorithm based on the three-dimensional Hough Transform.

Our data was gathered using a binocular stereo camera with a 4-cm baseline. Our input data set consisted of images of indoor office and hallway scenes with varying levels of clutter. Each entry in the data set consisted of an image pair: one, a monochrome image from the camera's left eye, and two, a "depth map" containing the estimated distance value for each pixel (provided directly by the camera software [1]). Figure 1 shows two example pairs of images. Dark blue regions in the depth map indicate points where no distance readings were returned due to lack of distinct features in that area.

One of our primary assumptions was that there were two types of evidence for planes: one, localized and contiguous evidence found on textured surfaces (such as desks), and two, sparse and scattered evidence when most of the plane is featureless and returns no depth data (such as walls and ceilings). Our algorithm focuses primarily on utilizing the first type of evidence since it was typically the most reliable. In regions where the second type of evidence predominates, our algorithm uses the mono image data to assist plane classification.

II. The Algorithm

A. Depth Map Processing

Before proceeding with computation, we observed that the stereo depth maps suffered from two main sources of noise: Gaussian noise on the distance readings and another source akin to "salt-and-pepper" noise which was caused by poor feature matching from the stereo camera (returning distance readings many meters off from the true values). To combat both of these sources of noise the depth map was subjected to

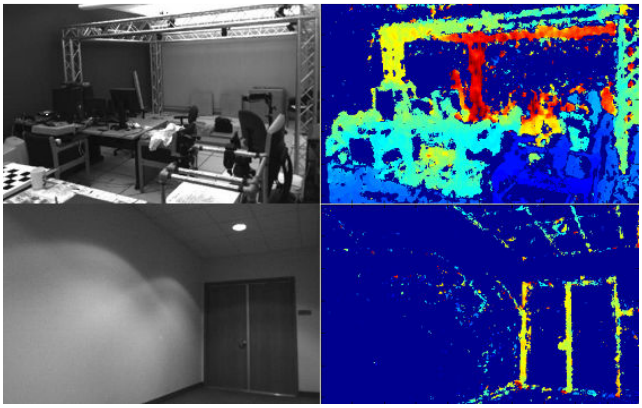


Fig. 1 Two example mono camera images (left side) and their corresponding depth maps (right side). Dark blue in the depth maps indicates places where no data was returned.

a b

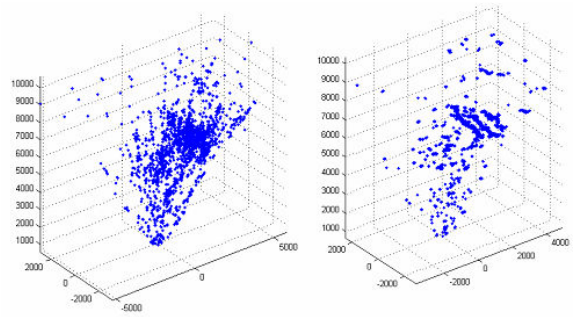


Fig. 2 Effects of modified low-pass filtering on the point cloud obtained by back-projecting the depth map shown on the lower right in Fig. 1. Here (a) is the original, unfiltered depth map and (b) is the filtered one. Note that the three vertical lines from the door are lost in (a) but appear strongly in (b).

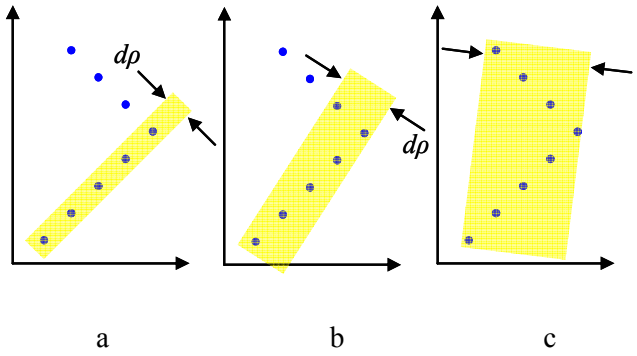


Fig. 3 Example of the sensitivity of the 3D Hough Transform to $d\rho$. The maximum response of the Hough Transform in each case is shown in yellow. In (a) $d\rho$ is chosen well and the maximum response selects a strong plane; in (b) $d\rho$ is slightly too large and the plane fit is off; and in (c) $d\rho$ is much too large and all planar information is lost.

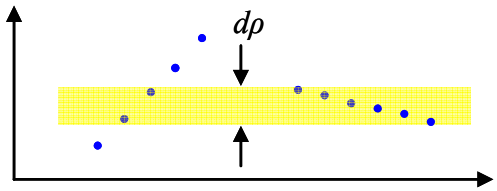


Fig. 4 Example of the maximum response of the Hough Transform being skewed by separate groups of planar points even when $d\rho$ is chosen well.

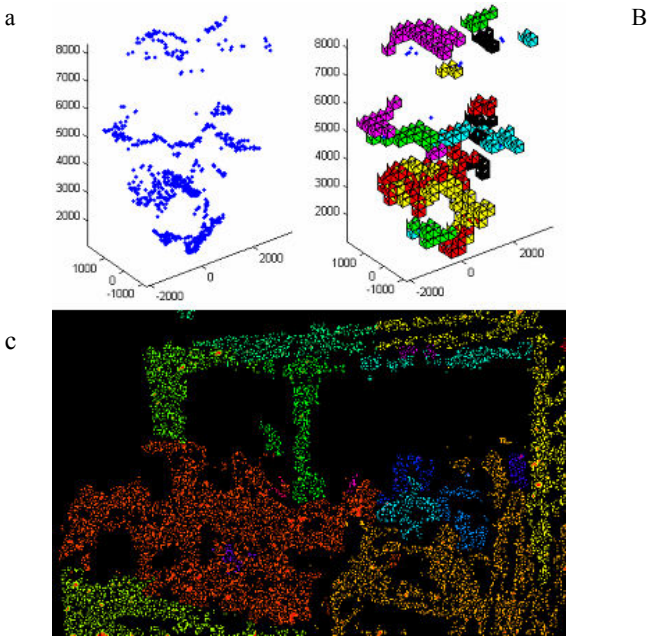


Fig. 5 The effects of segmenting the point cloud in (a) are shown as color-coded groups in (b). (c) is obtained by projecting the color-coded groups back to a 2D image.

modified low-pass filtering before it was used. After back-projecting the smoothed depth data into a 3D point cloud, the beneficial effects of this step can be seen in Figure 2.

The depth map was also decimated in order to reduce computation time. We used a modified random decimation algorithm where points closer to the camera were decimated more heavily than points farther away, thus roughly preserving the density of data points per surface area in the point cloud. In images with many data points, the decimation ratio was usually 10:1, but in sparse images no decimation was performed so as to retain all of the given data from the stereo camera.

B. 3D Hough Transform

Next the 3D Hough Transform is run on the 3D point cloud. In the Hough Transform, every point "votes" for every plane that passes within some distance of it. Thus the maximum response over the transform indicates the best guess for a plane in the region.

Planes in the 3D Hough Transform are described by their normal vectors which are specified by two angles (azimuth, θ, and elevation, φ) and the vector's Euclidean norm (ρ). Since the Hough Transform is discrete, it is parameterized by the step size in all three of these variables. Of the three of these, the Hough Transform's maximum response was most sensitive to $d\rho$ (as illustrated in Figure 3). We addressed this sensitivity by setting $d\rho$ to be 10 cm since this was small enough to detect most planes throughout the data set while still large enough to accommodate the noise. Step sizes of 5º proved sufficient for both $d\theta$ and $d\varphi$.

Another problem was that the Hough Transform's maximum response over several, unconnected groups of points was often non-planar even when the individual groups themselves were very planar (this is illustrated in Figure 4). We addressed this problem by first performing 3D segmentation on the point cloud and then running the Hough Transform on each of those segmented groups in isolation.

Our 3D segmentation algorithm works by quantizing the entire point cloud into a series of quantum boxes of size 30x30x30 cm. Next it selects the quantum box containing the highest number of points and connects to it all of the other contiguous quantum boxes also containing a high number of points. This group is then labeled and removed from the set of points. This procedure is repeated until 80% of the point cloud has been grouped (the remaining 20% were typically outliers). A sample result of this 3D segmentation is shown in Figure 5.

After this, the Hough Transform is run on each segmented group of points in the following manner. First, it is run over all of the points and the maximum response is determined (this is shown in Figure 6.a with the red points). Then all of the points corresponding to the maximum response are extracted and the Hough Transform is re-run on the remaining points (Figures 6.b and 6.c show these successive applications of the Hough Transform). This is repeated until 70% of the original points have been exhausted (processing the final 30% often generated many weak plane guesses).

C. Plane Guess Processing

After the Hough Transform step, the algorithm possesses a series of plane guesses specified by their normal vectors, centroids, and all of their assigned points. These raw results often contain many false planes so they are first decimated: those guesses that are drawn from too few depth points or

which occupy too little surface area are thrown away.

Also, because of the fineness of $d\rho$, these guesses often contain many repetitious guesses for a single plane; thus the guesses are clustered. In this step, two or more planes are combined if their normal vectors are highly aligned and if their centroids also satisfy some similarity conditions. Also factored in are elements from each point cloud's singular value decomposition which contained vital information on each plane's geometry.

We also made use of the requirement that valid planes do not occlude too many of the given stereo data points (we assume all planes are solid and opaque, so line-of-sight constraints must be satisfied). Thus, before any two plane guesses are combined, the supposed new plane is checked to see if it occludes too much depth data (this is illustrated in Figure 7). Single plane guesses are also dropped if they violate this occlusion rule themselves.

Figures 9.e and 10.e show the results after this step: colored regions indicate points in evidence for the final plane guesses, and the attached point cloud plots show the normal vectors of all plane fits.

D. Plane Region Labeling

Finally we desired to make the plane labels match the monochrome images better. The output labels of the last step did not match very well because they were drawn entirely from the stereo data and so had little direct relation to the mono image boundaries. We corrected this by introducing image segmentation on the mono images. We used a super-pixel-, graph-based approach published by Felzenszwalb and Huttenlocher [2]. An example output of their segmentation algorithm is shown in Figure 8.

For the segmented images, we adjusted the parameters so that the segmentation was fine enough that each segment overlapped with only one final output plane with high probability. Thus, in the final step of the algorithm, each image segment is assigned to the plane label with which it has the most overlap (examples of these processed images appear in Figures 9.c and 10.c). This has the effect of spreading out the final plane labels into locations better-defined by the image boundaries, and it also allows a small amount of depth data to provide a plane label for large, texture-less regions that returned no data.

III. RESULTS

Figures 9 and 10 depict some representative results. For reference, images showing the results on all of our input data are attached to this report.

Figure 9 shows a typical result on a textured, close-up image. Our algorithm performs best on these types of images since the planes are highly-textured and are close to the camera, producing dense point clouds and enabling very reliable results.

Figure 10 shows a result on a cluttered indoor scene with most objects lying farther away from the camera than in Figure 9. Here the algorithm still succeeds at finding most of the

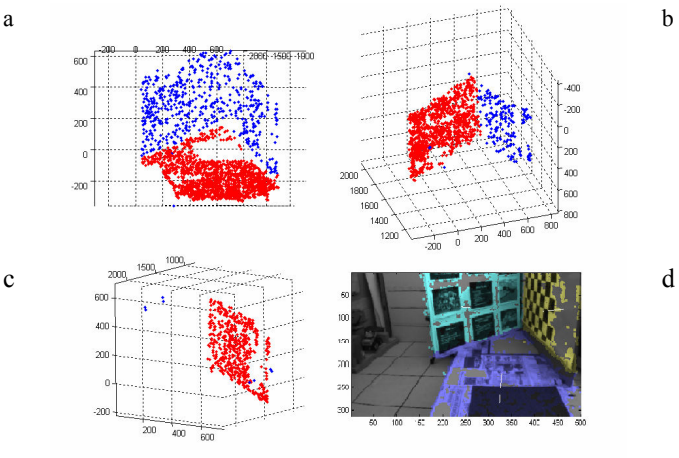


Fig. 6 Iterative procedure for running the Hough Transform. (a) shows the first maximum response; (b) shows the next maximum response after the points from the first have been removed; and (c) shows the last maximum response. (d) shows these three plane labels on the corresponding mono image.

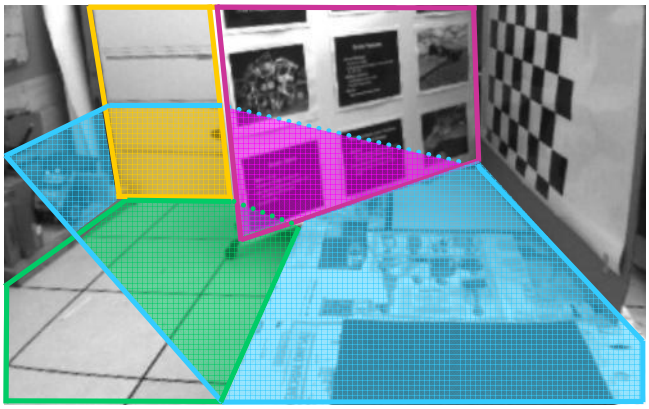


Fig. 7 Plane occlusion example. Here the algorithm considers combining the two separate light blue groups into one plane. It calculates the convex hull (blue outline) of the new, proposed plane and checks whether it occludes any points. In this example, the purple points lie in front of the plane and are not occluded, but the yellow and green points lie behind the plane and are occluded. Thus the two blue planes are not combined into a single plane.

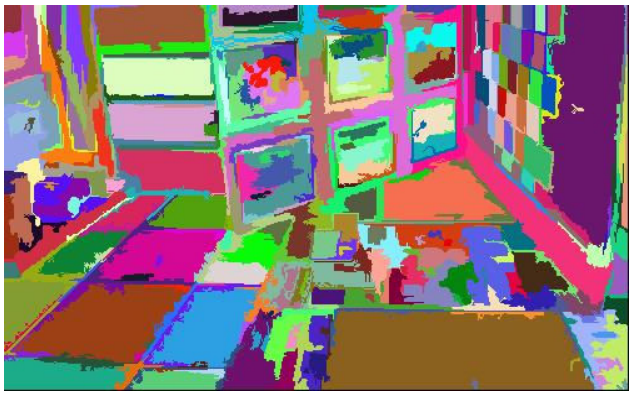


Fig. 8 Sample output of the image segmentation algorithm by Felzenszwalb and Huttenlocher [2]. Parameters set to k=500, σ=0.3.

planes but it has trouble assigning them to the proper image regions. It also exhibits many more spurious results than in the previous example.

IV. Conclusions and Future Work

Overall our algorithm succeeded in finding many planes in certain settings. It works best in scenes with high-texture and low clutter where it is able to identify both the correct planes and their orientations with high accuracy. It is still weak in finding occluded planes in cluttered environments as well as in low-texture images with very sparse depth data. And even when it finds the correct planes it is still prone to mislabeling them in the output step.

The results achieved so far are promising, but there are many possible directions for improvement. For instance, our algorithm produces hard classifications, but another approach would be to build a probabilistic model that estimates the number of planes through an adaptive method and then assigns weights to each point indicating how likely they are to appear in each plane.

One could also employ prior assumptions about the geometry of the room to reduce spurious results. Some possible assumptions include that there are always walls bounding the scene, that the major planes should be orthogonally oriented (the walls, floor, and ceiling), and that the best planes tend to be strictly horizontal or vertical (table surfaces, desk tops, and doors).

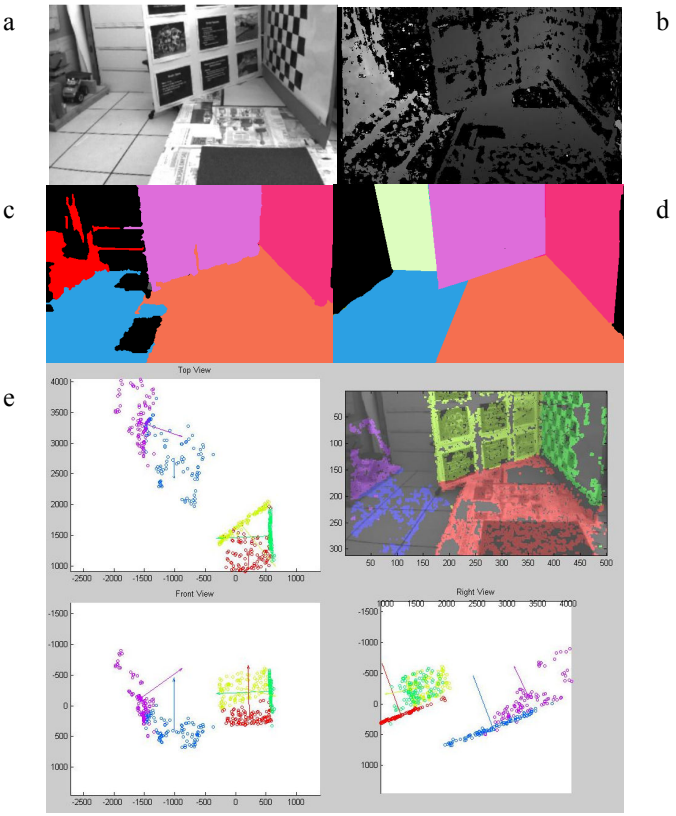


Fig. 9 Example results. (a) shows the original mono image, (b) the original depth map, (c) the process and labeled output, (d) the hand-labeled ground truth, and (e) a composite image showing the points for each final plane as well as their normal vectors.

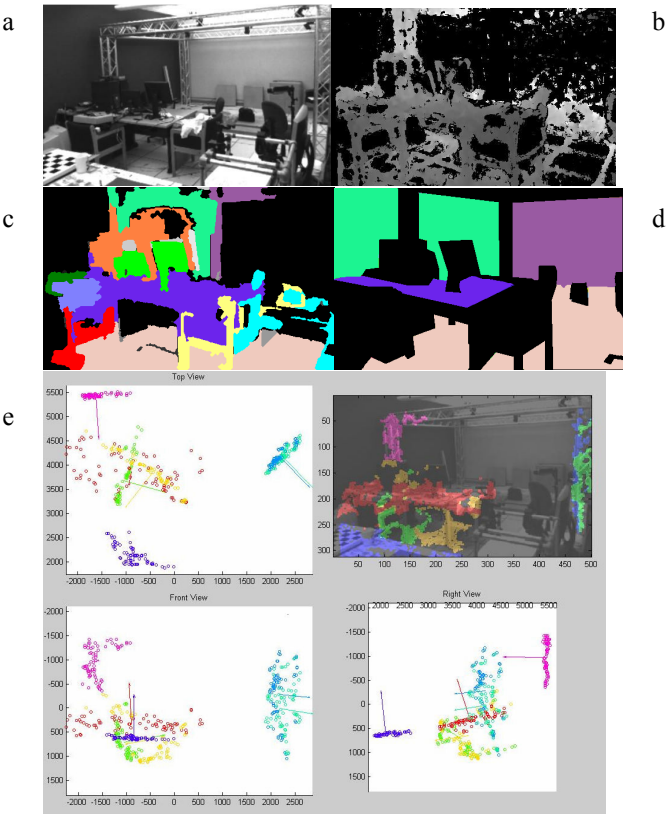


Fig. 10 Another set of example results with the same image arrangement as given in Figure 9. Here the scene contains more clutter and is a more open shot.

Lastly, we only examined the task of finding planes given a single image, but in a mobile robot application it would be possible to make use of information from multiple, adjacent video frames when attempting plane classification.

Acknowledgments

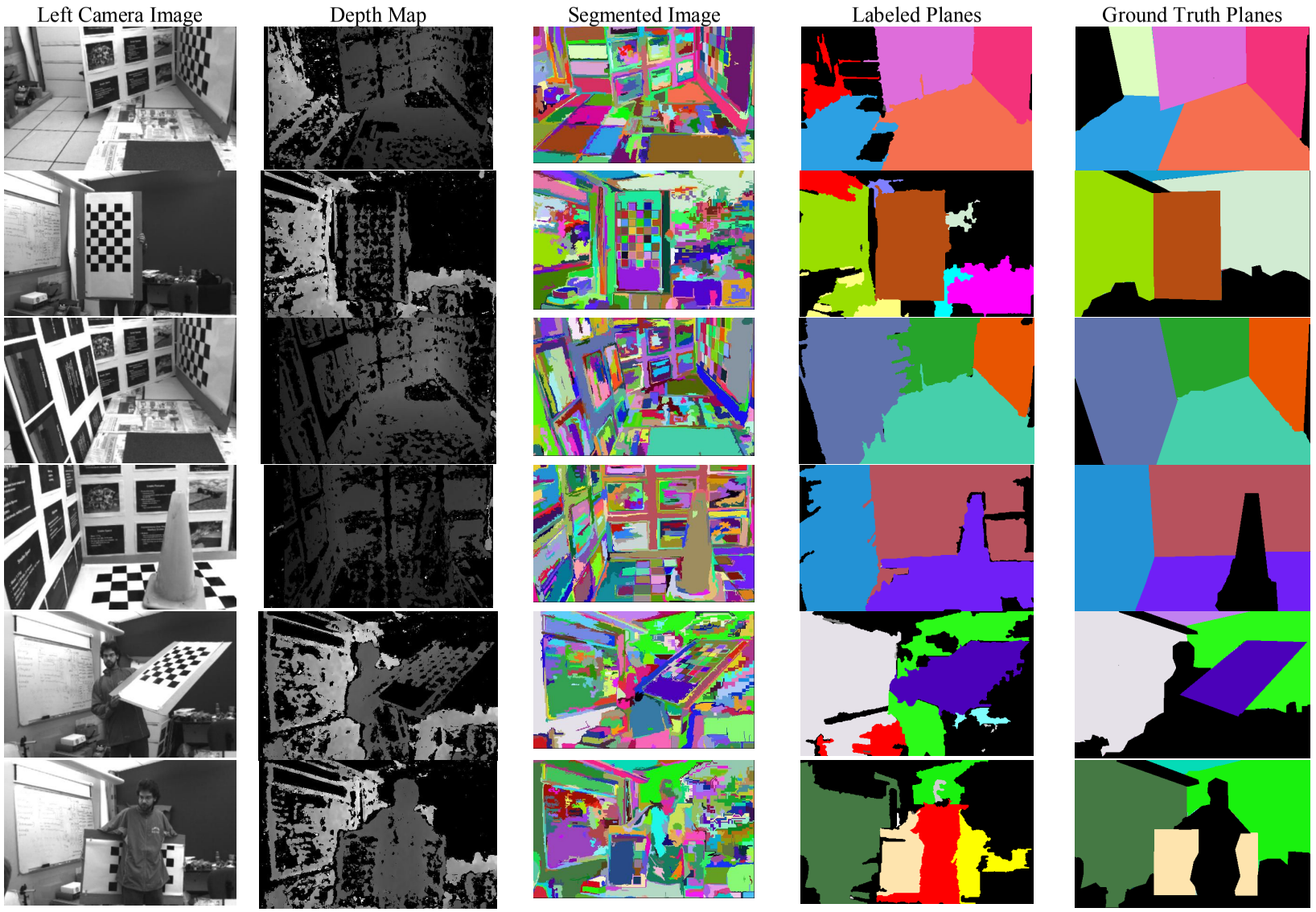
We would very much like to thank the many people who provided advice and direction on this project, including Prof. Andrew Ng, Prof. Jana Kosecka, Stephen Gould, Ashutosh Saxena, and the members of the Fall 2006 STAIR Vision team.

References

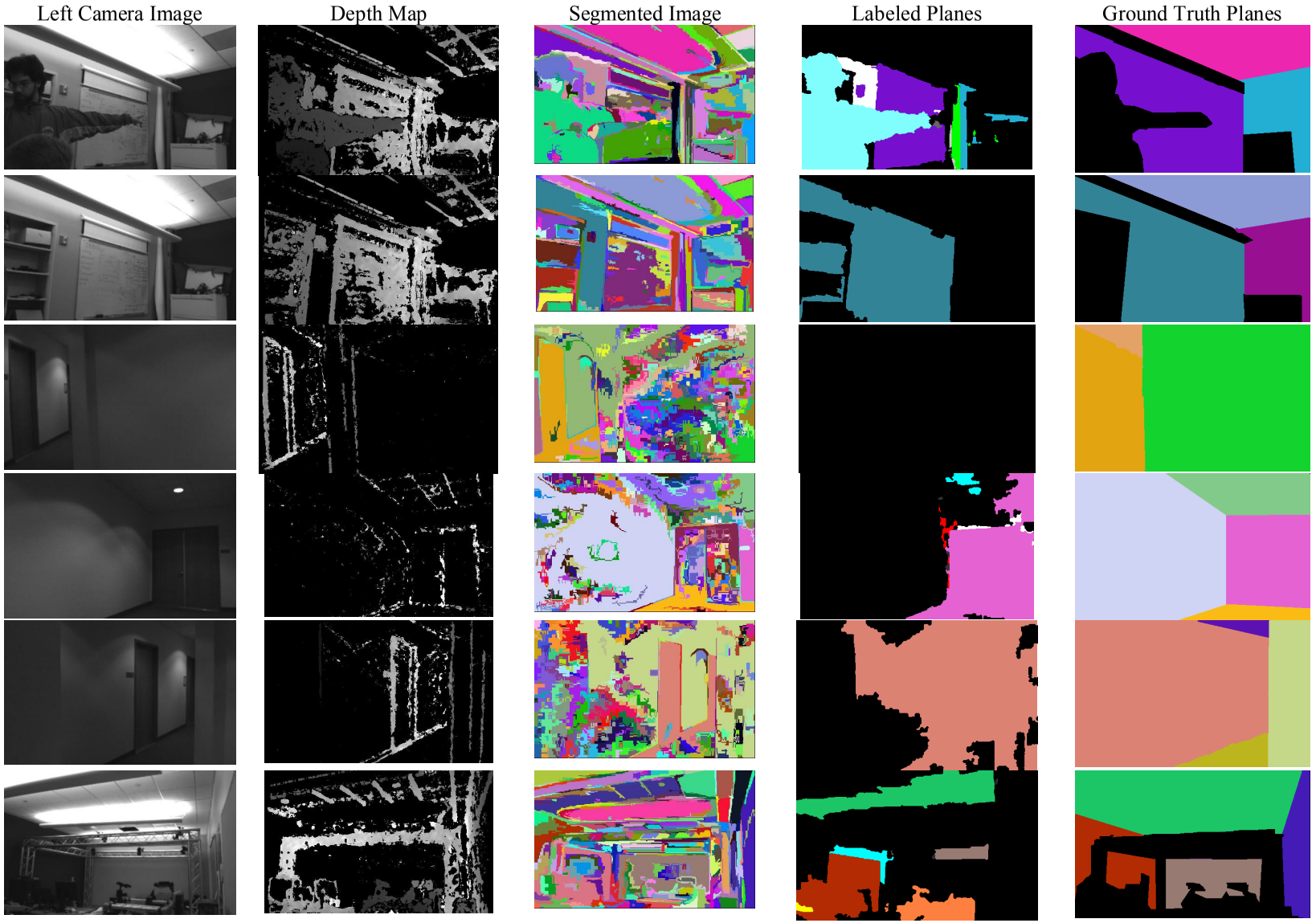
[1] Tyzx stereo camera information. http://www.tyzx.com/products/index.shtml

[2] P. F. Felzenszwalb and D. P. Huttenlocher. "Efficient Graph-Based Image Segmentation" International Journal of Computer Vision, Volume 59, Number 2, September 2004. http://people.cs.uchicago.edu/~pff/segment/

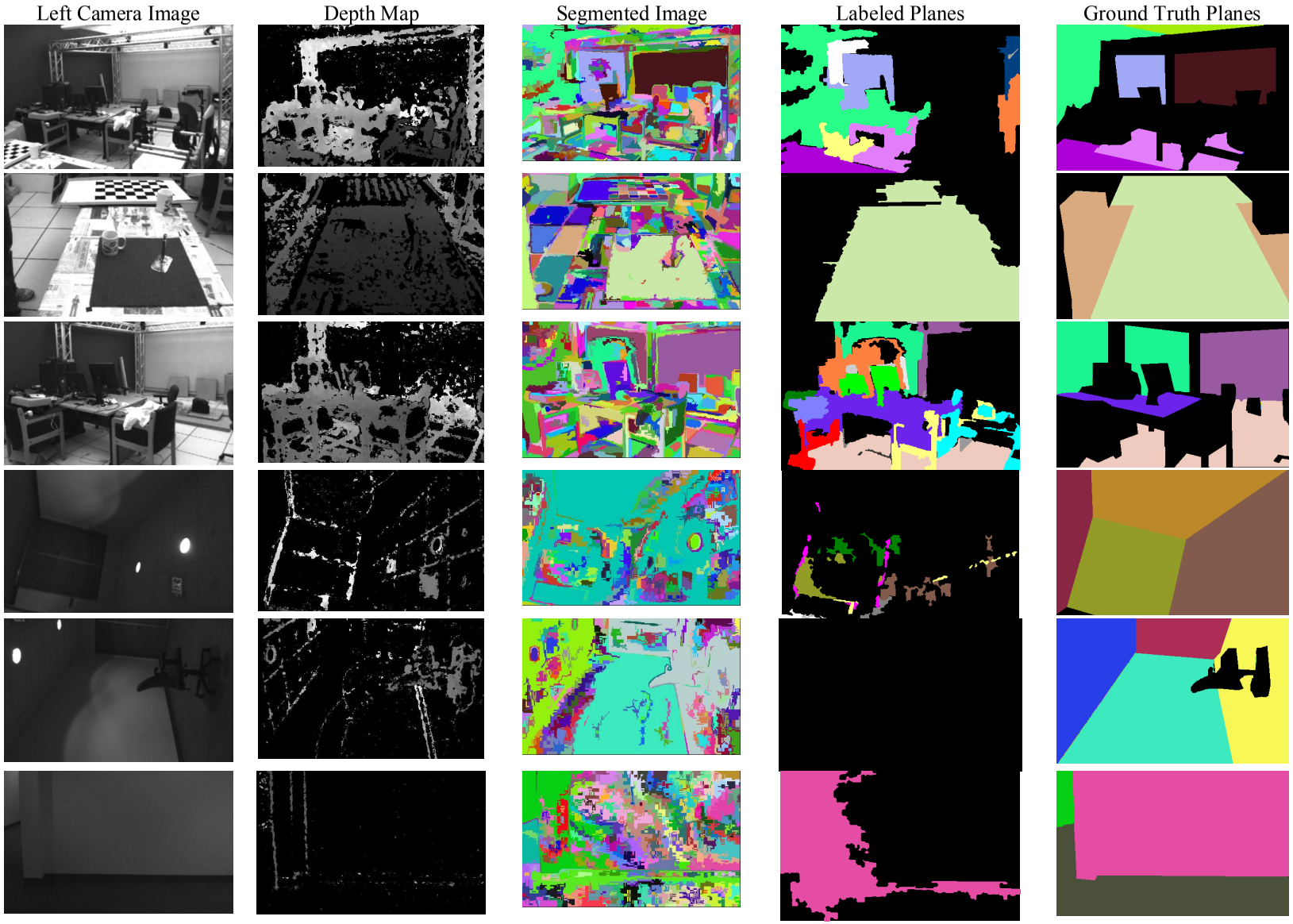
Appendix


Left Camera Image
Depth Map
Segmented Image
Labeled Planes
Ground Truth Planes

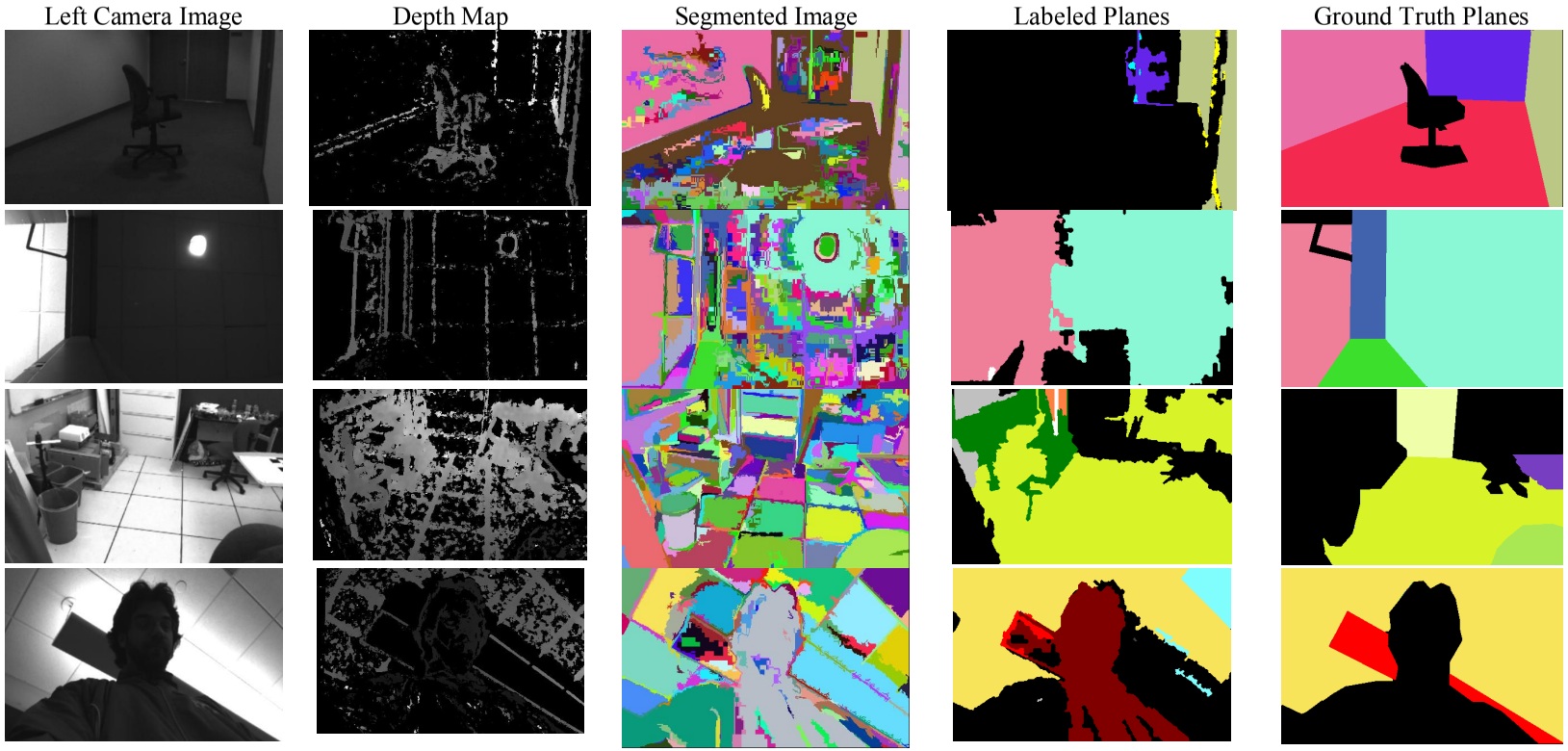
Appendix


Left Camera Image
Depth Map
Segmented Image
Labeled Planes
Ground Truth Planes

Appendix


Left Camera Image
Depth Map
Segmented Image
Labeled Planes
Ground Truth Planes

Appendix


Left Camera Image
Depth Map
Segmented Image
Labeled Planes
Ground Truth Planes