# UNIVERSAL OBJECT SEGMENTATION IN FUSED RANGE-COLOR DATA

by

Jeffery Michael Finley

B.S., Kansas State University, 2006

---

A THESIS

submitted in partial fulfillment of the
requirements for the degree

MASTER OF SCIENCE

Department of Electrical and Computer Engineering
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas
2008

Approved by:

Major Professor
Dr. Chris Lewis

# Copyright

# Abstract

This thesis presents a method to perform universal object segmentation on fused SICK laser range data and color CCD camera images collected from a mobile robot. This thesis also details the method of fusion. Fused data allows for higher resolution than range-only data and provides more information than color-only data. The segmentation method utilizes the Expectation Maximization (EM) algorithm to detect the location and number of universal objects modeled by a six-dimensional Gaussian distribution. This is achieved by continuously subdividing objects previously identified by EM. After several iterations, objects with similar traits are merged. The universal object model performs well in environments consisting of both man-made (walls, furniture, pavement) and natural objects (trees, bushes, grass). This makes it ideal for use in both indoor and outdoor environments. The algorithm does not require the number of objects to be known prior to calculation nor does it require a training set of data. Once the universal objects have been segmented, they can be processed and classified or left alone and used inside robotic navigation algorithms like SLAM.

# Table of Contents

# List of Figures

# Chapter 1

# Introduction

The ability for a robot to detect and identify objects in its environment has several benefits. One such benefit is where objects detected by a robot can aid in map-building and navigation. A mobile robot's environment and the data it collects, both from cameras and range sensors, is inherently three-dimensional. The logical evolution of data processing must be from two to three dimensions. Image data becomes 3D when combined with motion measurements or when combined with laser range data. One widely used range scanner for mobile robots is the SICK LMS 200, which was originally designed for industrial automation applications.

Most research into 3D data processing and feature extraction is limited to range data or color data independently. However, there is an ever-growing interest in data fusion, the merging of laser range data and color imagery to create a single, three-dimensional, colorized dataset. Range and color complement each other nicely. Typically, image processing techniques suffer from their inability to ignore effects from shadows or lighting changes. While on the other hand, range data cannot distinguish between two identical objects that are colored differently (a door and a wall, for example). This thesis proposes a method for creating three-dimensional, colorized data points. With fused range and color data, this thesis describes a method for segmenting and subsequently identifying objects from that data. The proposed segmentation techniques rely on the Expectation Maximization algorithm. One advantage of this method is that it performs well in both indoor and outdoor environments. The reason for this is the use of a six-degree Gaussian distribution as a universal object model. This model accurately segments walls, furniture, and even trees and bushes. This presents an advantage over most segmentation algorithms that are designed for very specific environments or those that must maintain a database of object parameters in memory.

This thesis is organized into five chapters. This first chapter will continue with a discussion on the prior work. Chapter 2 will present several important concepts that must be understood before presenting the research. Beginning in Chapter 3 and continuing through Chapter 4 will be a discussion of a merging algorithm for range and color data followed by the segmentation algorithm. In order to provide continuity, all chapters contain figures from the same environment. This allows the discussion in each chapter to flow more smoothly and allow the reader to understand the entire process. The end of Chapter 4 contains an

analysis for the specific environment followed by Chapter 5 which summarizes the thesis and recommends future research.

## 1.1 Previous Work

There are a number of people who have developed methods for feature detection in range data. Pu and Vosselman scanned the outside facade of buildings and used the data to detect features like doors, windows, and walls[1]. Range points were first clustered using a surface growing algorithm. Then, each surface was tested against several criterion to determine if it was part of the ground or a door, for example. Another paper by Vosselman et al. introduced voxel space to filter range points[2]. Voxel spacing is simply discreetizing 3-space into bins and marking a "1" if a range point falls inside the bin and "0" if no points fall inside. From this voxel space, the authors note that several filters usually used on 2D image space, like the Hough transform, can easily be adapted to work in 3D voxel space. Using a 3D Hough transform, the authors extracted geometric shapes like cylinders, spheres, and boxes from range data in order to produce computer models of industrial environments and city landscapes. Instead of parameterizing geometric shapes, Biswas et al. kept "isosurfaces" of previously scanned objects[3]. The isosurfaces were created using radial basis functions about 500 pivot points. Subsequent scans were then compared, using spherical decomposition, against a database of previously created isosurfaces to find similarities. Biswas et al. note one benefit of spherical decomposition is rotational invariance. Isosurfaces provide a way to efficiently model complex shapes such as organic entities.

The fusion of range and camera data is attracting a growing body of researchers. A large portion of this research is devoted to finding better fusion techniques rather than feature extraction from the fused data. Teams like Abmayr et al.[4], Sequeira et al.[5], Nyland[6], and Dias et al.[7] fall into this category. All use high resolution laser scanners along with mega-pixel cameras. The goal of these authors is to create accurate 3D models of real-world environments for such subjects as historical preservation, city planning, and high-resolution map building. Such high resolution data makes further processing for robotic vision computationally expensive.

Some authors use a hybrid of range and color data. Pang et al. for example, use two cameras and a laser range finder in their research into feature detection and tracking[8]. Unlike the papers listed above, the laser scanner is only used to improve accuracy of range data previously computed from images and does not scan the entire scene. Thrun et al. produced world models by processing only range data, but then superimposed images taken from a camera onto that model[9]. Biber also only processed the range information before mapping images[10]. Both authors were mapping indoor environments where planar objects define the majority of the scene.

The paper "A Real-Time Expectation-Maximization Algorithm for Acquiring Multiplanar Maps of Indoor Environments With Mobile Robots" was the inspiration for using the Expectation Maximization (EM) algorithm to locate objects in three-space[9]. The authors achieved this by maximizing the expection of the data using planar object models. Thrun

et al. also developed algorithms to evaluate how "good" a plane fit was. Planes that were considered "weak" were pruned while similar planes were merged. This pruning and merging between iterations of the EM algorithm proved to be an effective way to discriminate between poorly defined mixtures. Also, this meant that the number of models maximized by EM was not constant. Therefore, Thrun not only segmented the data into planar models which maximized the expectation of the data, but also detected the number of planes in an environment without proior knowledge. For indoor environments, a planar model performs very well. Man-made objects are commonly planar and most indoor environments consist mainly of walls. For example, in one indoor data set, Thrun et al. report that almost 95% of all the measurement points could be accuratly ascribed to planar features. For data that is not accurately defined by a plane, their algorithm leaves the data as a fine-grained polygonal model. While this method works very well for highly planer environments, it will not perform well in outdoor settings were geometric, man-made structures are sparse.

Another influential work is that of Ueda, Nakano, Gharamani, and Hinton[11]. The authors present an Expectation Maximization algorithm along with a split and merge criteria to locate and parameterize two-dimensional Gaussian densities. The authors remark that their solution helps solve problems when many Gaussian densities occupy one region while other regions contain relatively few Gaussian densities. The criteria for splitting and merging were simple, yet effective. Ueda et al. described split values and merge values that are used to represent how accurately points are defined by a density. First, each density receives a split value and a pair of densities receive a merge value. Next, the density with the lowest split value is split while the density pair with the highest merge value is merged. These two calculations are made after every M-Step. A partial E-Step is run once the mixtures have been reorganized. If the splitting and merging has yielded a better likelihood, the new mixtures are kept. If they did not yield a better likelihood, the new mixtures are discarded and EM iterates. It may be noted, that every time a split occurs, a merge occurs. This kept the number of mixtures maximized by EM the same, which facilitated comparison of the likelihood. This also means the number of mixtures must be known or determined prior to execution.

# Chapter 2

# Background Information

## 2.1 Coordinate Transforms

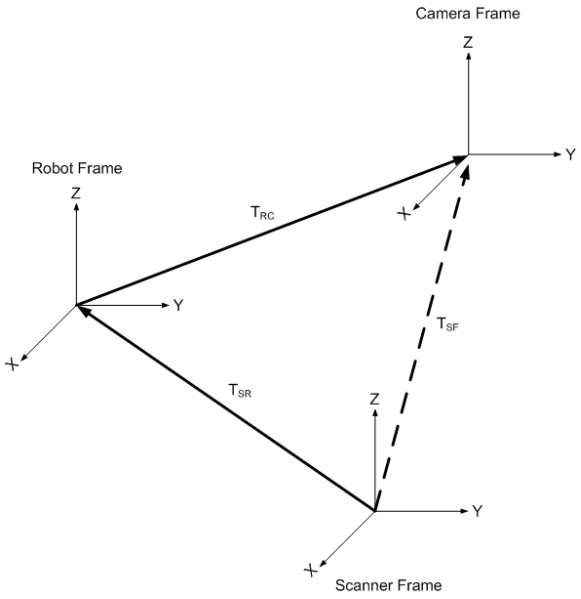



Figure 2.1: Different coordinate frames and the transforms ($T_{SR}$, $T_{RC}$, $T_{SF}$) that define them.

Every measurement device records data in its own frame of reference. Figure 2.1 shows three different coordinate axes for three different frames of reference: the laser scanner, the robot, and camera. For data fusion, all data must be expressed in the same frame of reference. Coordinate transforms, $T_{ij}$, relate two frames of reference and allow data to be expressed in a different frame than which it was gathered. Homogeneous coordinate transforms are commonly used for this task. A homogeneous transform allows translations and rotations through matrix multiplication. For example, to express a point $\overline{p_1}$ defined in $frame_1$'s frame of reference in $frame_2$'s frame of reference, $T_{12}$ must be defined. Using $T_{12}$

we can find $\overline{p_2}$, $\overline{p_1}$ as seen by $frame_2$, by $\overline{p_2} = T_{12} \cdot \overline{p_1}$. If $frame_1$ and $frame_2$ only differed in translation $(\Delta x, \Delta y, \Delta z)$, the transform would be:

$$\overline{p_2} = T_{12} \cdot \overline{p_1}$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \Delta x \\ 0 & 1 & 0 & \Delta y \\ 0 & 0 & 1 & \Delta z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

If $frame_1$ and $frame_2$ differed by a rotation of $\theta$ about the x-axis, a rotation of $\psi$ about the y-axis, and a rotation of $\phi$ about the z-axis, $T_{12}$ could be broken up into three matrices:

$$\overline{p_2} = T_{12} \cdot \overline{p_1}$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) & 0 \\ 0 & \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos(\psi) & 0 & \sin(\psi) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\psi) & 0 & \cos(\psi) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos(\phi) & -\sin(\phi) & 0 & 0 \\ \sin(\phi) & \cos(\phi) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

The relationship between any two coordinate systems can be expressed by six parameters. These are the translations ($\Delta x$, $\Delta y$, $\Delta z$) and rotations ($\theta$, $\psi$, $\phi$). Each parameter defines a homogeneous transform, and the product of the individual transforms defines the overall coordinate transform.

## 2.2 Camera Calibration

The most basic model for a camera is the pinhole camera model. The pinhole camera model describes the relationship between a 3D point and its 2D projection onto an image plane. In this model, light rays pass through a single point to form a perfect image. Real-world cameras produce distorted images. A calibration procedure seeks to determine the distortion parameters intrinsic to the real camera and lens. Using these parameters, images from the camera can be rectified to create undistorted images as if from an ideal pinhole camera. The parameters defining the relationship between the camera's frame of reference and the world coordinate frame are termed the extrinsic camera parameters.

### 2.2.1 Intrinsic Parameters

There are several sources of image distortion with imaging systems. Typical camera lenses radially distort images. Radial distortion is usually seen as a bowing of physically straight lines in the image as can be seen in Figure 3.2a. Another cause of distortion for digital cameras is the CCD array, which is made up of discrete pixels, unlike the pinhole camera

model that has infinite resolution. Also, the pixels themselves may not be square, or they may not be equally spaced vertically and horizontally. The toolbox by Strobl et al. provides MATLAB code for determining a camera's intrinsic distortion parameters and subsequent code for rectifying its images[12].

The intrinsic camera parameters used in Strobl et al. are: focal length $[fc]_{2x1}$, principal point $[cc]_{2x1}$, skew coefficient $alpha\_c_{1x1}$, and distortion coefficients $[kc]_{5x1}$[12]. If $fc(1)/fc(2) \neq 1$, then the camera's CCD array is not square. The principal point of an image is the location with the least amount of distortion, ideally this would be the center of the image, but in practice is usually slightly off-center. Skew and image distortion coefficients are the coefficients used inside a lens model. A brief description of the lens model used by Strobl follows.

Given a pixel location $p_d = \begin{bmatrix} x \\ y \end{bmatrix}$ and $r : r^2 = x^2 + y^2$, the normalized point coordinate is

$$x_d = (1 + kc(1)r^2 + kc(2)r^4 + kc(5)r^6)p + dx$$

where $dx$ is the tangential distortion vector

$$dx = \begin{bmatrix} 2kc(3)xy + kc(4)(r^2 + 2x^2) \\ kc(3)(r^2 + 2y^2 + 2kc(4)xy \end{bmatrix}$$

The final pixel location, taking into account lens distortion, is

$$p_{ud} = \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} fc(1)(x_d(1) + alpha\_c * x_d(2)) + cc(1) \\ fc(2)x_d(2) + cc(2) \end{bmatrix}$$

### 2.2.2 Extrinsic Parameters

For this research, the extrinsic parameters for the camera define the relationship between the camera's reference frame and the laser range scanner's reference frame. These parameters determine the homogeneous transforms used to express the range data in the camera's frame. The parameters are expressed in translational differences in $x$, $y$, and $z$, plus angle's $\phi$, $\theta$, and $\psi$.

## 2.3 The Expectation Maximization Algorithm

The Expectation Maximization (EM) algorithm is used to estimate the model parameters, $\Omega$, for a set of densities $f(z|\Omega)$ from a given dataset $Z$, such that $g(Z|\Omega) = \int_i f(z_i|\Omega_j)dx$. The EM algorithm iterates through two steps, an expectation step (E-Step) and a maximization step (M-Step). Iteration continues until the likelihood function has been maximized. It is important to note that, for a given data set, $Z$, there are global and local maxima's. EM does not necessarily converge to the global maximum but will, however, always converge.

### 2.3.1 Log-Likelihood Function

A likelihood function in mixture modeling describes the likelihood that a collection of sampling densities $\Omega_J$ will produce the dataset $Z$. It is this function that EM maximizes by iterating through two steps; an E-Step and a M-Step. This thesis, like Bouman[13] and Ueda[11], uses Gaussian densities whereas Thrun[9] used a Gaussian noise model for planar objects. The equation for a (univariate) Gaussian density function is:

$$p(z_i|\Omega_j) := \pi_j * \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(z_i-\mu)^2}{2\sigma^2}} \quad where \quad \sum_j \pi_j = 1 \tag{2.1}$$

where $\Omega_j$ are the model parameters: mean $\mu$ and variance $\sigma^2$. Individual expectations for each measurement $z_i$ are then multiplied to create the likelihood function for the entire dataset.

$$likelihood = \prod_i \sum_j \frac{\pi_j}{\sqrt{2\pi\sigma_j^2}} e^{-\frac{(z_i-\mu_j)^2}{2\sigma_j^2}} \tag{2.2}$$

Taking the logarithm of 2.2 produces a log-likelihood function. Log-likelihoods maintain all the extrema of the original likelihood, but are easier and quicker to calculate.

$$log\text{-}likelihood = \sum_i \sum_j \log\left(\frac{\pi_j}{\sqrt{2\pi\sigma_j^2}}\right) \frac{(z_i-\mu_j)^2}{2\sigma_j^2} \tag{2.3}$$

### 2.3.2 Initialization

The EM algorithm requires a starting number of densities and their parameters at initialization. The initialization used by Bouman[13] will be explained here. The first step is always to determine how many densities, $J$, we wish EM to maximize. For $J$ means, Bouman selects $J$ evenly spaced points from the dataset. Each density begins with the same variance. Bouman uses the variance of the entire dataset.

$$\sigma^2 = \frac{1}{I-1} \sum_{i=1}^{I} (z_i - \mu)^2$$

### 2.3.3 E-Step

The purpose of the E-step is to determine the expectation of the data given the current model of the world. That is, if the world was made of $J$ densities, then for each data point $z_i$ there is a certain expectation $p(z_i|\Omega_j)$ that could be attributed to each of the densities. The log-likelihood is calculated here.

### 2.3.4 M-Step

Each iteration of the M-Step maximizes the models' parameters, $\Omega_j$, using all $z_i$'s. There are several methods used to maximize model parameters, the most popular of which is the Maximum Likelihood Estimator, or MLE. For $N$ discrete samples in a Gaussian distribution, the M-Step proceeds as follows:
First, calculate the new mean.

$$\bar{\mu_j} = \frac{\sum_{i=1}^{N} z_i * p(z_i|\Omega_j)}{\sum_j p(z_i|\Omega_j)} \tag{2.4}$$

Second, calculate the new variance.

$$\bar{\sigma_j} = \frac{\sum_{i=1}^{N} \left(z_i - \bar{\mu_j}\right)^2 * p(z_i|\Omega_j)}{\sum_j p(z_i|\Omega_j)} \tag{2.5}$$

Adjust the mixing weights.

$$\pi_j = \frac{1}{N} \sum_{i=1}^{N} p(z_i|\Omega_j) \tag{2.6}$$

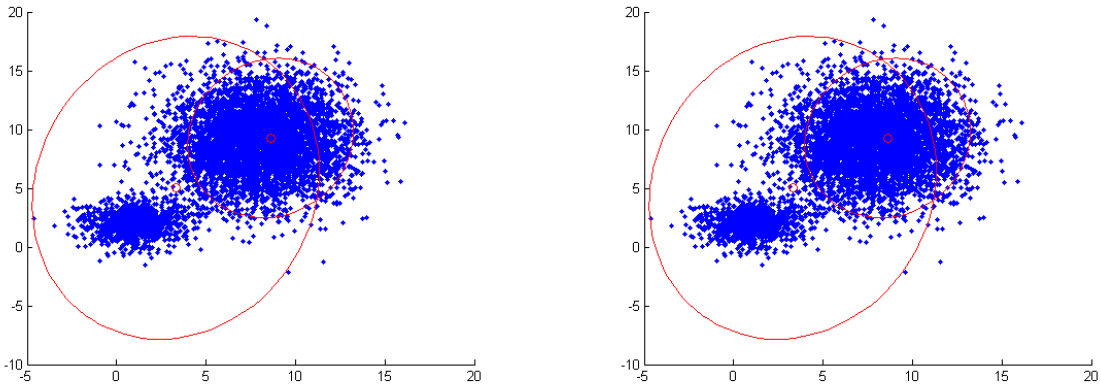
Continue for all $J$ models.

### 2.3.5 Iteration

As stated previously, EM is iterative in nature. The initialization "guesses" are used for the model parameters in order to prime the algorithm for its first iteration. For every iteration, the expectation of each measurement $p(z_i|\Omega_j)$ is calculated for all $J$ densities in the E-Step. Then the model parameters $\Omega_j$ are maximized in the M-Step. Finally the log-likelihood function is re-calculated using the appropriate $\Omega_j$ that maximizes $p(z_i|\Omega_j)$ for each $z_i$. Iteration continues until the log-likelihood either stops increasing, or slows dramatically.

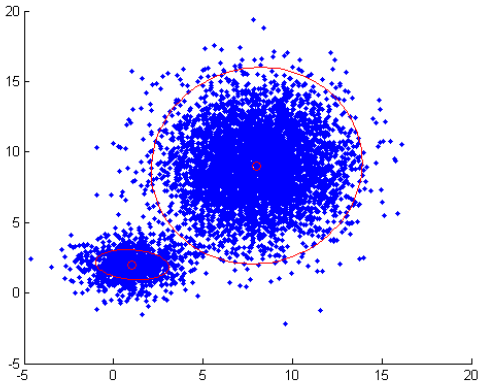
### 2.3.6 Two-Dimensional Example

As an example of EM applied to a data set, a collection of points were generated from two Gaussian densities. The EM algorithm will search for the two densities that maximize the probability of the data set being generated from those densities. Figure 2.2 displays how the EM algorithm adjusts the parameters of the two Gaussian densities throughout the iterations. Note how the means approach the center of each cluster and the variances reduce throughout the iteration.




(a)

(b)




(c)

Figure 2.2: The densities are initialized in (a), (b) shows the densities after three iterations, (c) displays the final parameters.

# Chapter 3

# Merging Range and Color

## 3.1 Data Collection

Image and range data was gathered from different locations using the mobile robot in (3.1). This robot runs a version of Pebble Linux. Mounted on the robot is a laser scanner and a color camera. Both devices are mounted on a tilt unit. When referring to a tilt angle, 0 ° will be defined as the angle when the laser scanner is level with the ground. The author developed code for the robot to collect laser and camera data. During the code's execution, the scanner begins at a tilt of −30 ° and increments up to 60 °, collecting one laser scan and taking one image every 1 °.

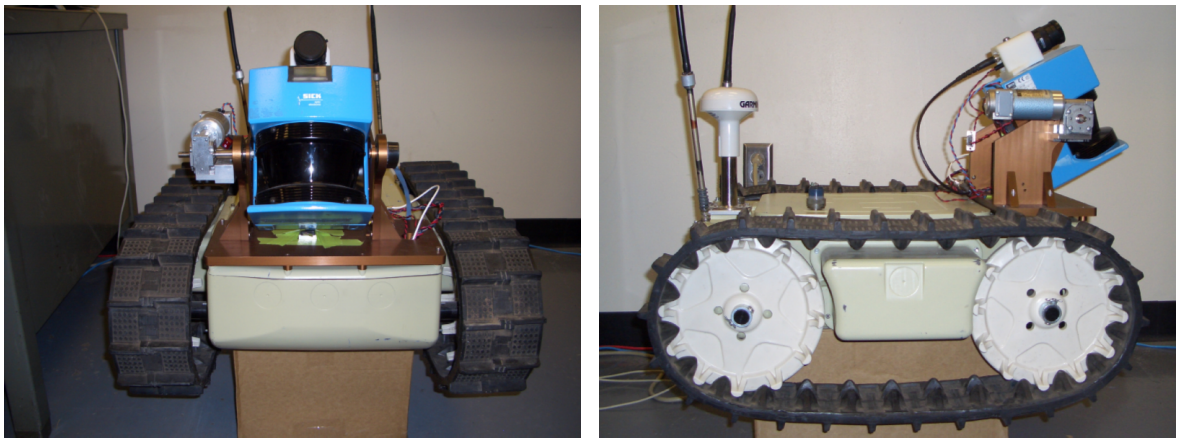


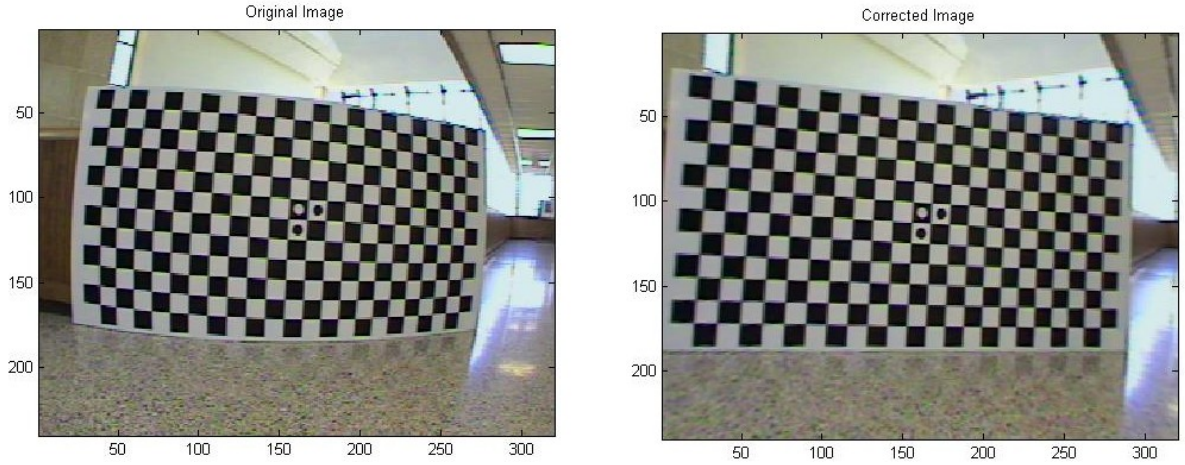
Figure 3.1: The robot used to take the data. The laser scanner is a SICK LMS 200 with the resolution set to 1 °. The camera is a Watec WAT250D color camera with a Computar T3Z3510CS lens. Fused laser and range data is collected and processed for mapping and navigation.

### 3.1.1 Camera

The camera is a Watec WAT250D color camera with a Computar T3Z3510CS lens. The resolution of each image is 240 pixels by 320 pixels by the time the camera data is read into MATLAB.

#### Image Calibration

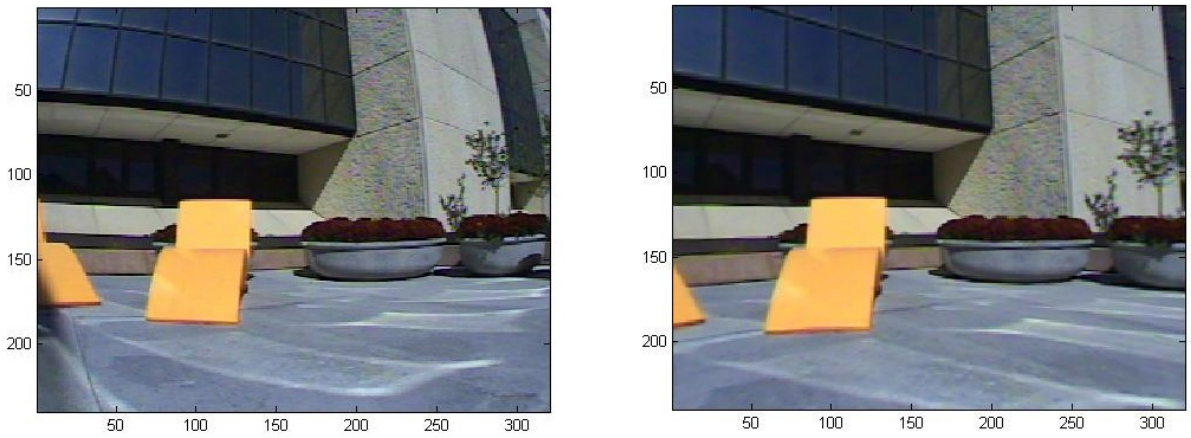
Strobl et al. provides a inter-active MATLAB program to calibrate a camera image for distortion called `calib_gui.m`[12]. This program requires several shots of a well-defined checkerboard. After a number of steps, the calibration routine returns several parameters: focal length $[fc]_{2x1}$, principal point $[cc]_{2x1}$, skew coefficient $alpha\_c$, and distortion coefficients $[kc]_{5x1}$. The parameters are fed into the function `rect(I,R,fc,cc,kc)` also by Strobl et al. that rectifies the given image `I`. The variable `R` is a $3x3$ identity matrix.




(a) (b)

Figure 3.2: A picture of the calibration grid before (a) and after (b) correcting for distortion. Note the board and checkers are rounded in the un-calibrated picture.

As previously stated, extrinsic camera parameters deal with a camera's location in relation to the laser scanner. Both scanner and camera are rigidly mounted. The author measured a vertical (z-axis) and horizontal (x-axis) distance between the location of the laser receiver and the camera's CCD array. These values were entered into a translation matrix and used to transform laser data into the camera's frame. The measured values were later adjusted to yield a more accurate fusion. Both the laser scanner and camera were initially assumed to have no rotational alignment differences, but these values were also experimentally adjusted to produce a more accurate fusion.




(a) (b)

Figure 3.3: A picture taken with a tilt angle of 0° before (a) and after (b) correcting for distortion.

#### Projecting Pixels

Images are two dimensional representations of an inherently three dimensional world. With the pin hole camera model, each pixel defines a vector, $\overline{v}$ starting as the pixel on the focal plane passing through the focal point and projecting out towards infinity. A convienent equivalent camera model reverses the pin hole and focal plane so that the image is not inverted. An image feature's world location lies somewhere on that vector. The process of re-projecting that vector attempts to more accurately constrain the image vector to a specific length. If done correctly, each vector end on the surface of an object in three-space that produced that pixel information.

### 3.1.2 Laser Scanner

The laser scanner is a SICK LMS 200. The scanner uses a rotating mirror to obtain range measurements in 1° increments. The range is measured in centimeters with a maximum range of 8000 cm. The laser scanner has a field of view of 180°. The author has defined the range measurement taken directly in front of the robot to be at 0°.

#### Coordinate Transformation

All range points are measured with reference to some point within the laser scanner head. The SICK system produces laser scanner data in a spherical coordinate system $(z, \phi, \theta)$, where $z$ is the range measurement, $\phi$ is the scan angle, and $\theta$ is the tilt angle. After a complete cycle, the range data is located in a 81x181 array where each row represents 1° of tilt and each column represents a 1° change in scan location. The range data output file

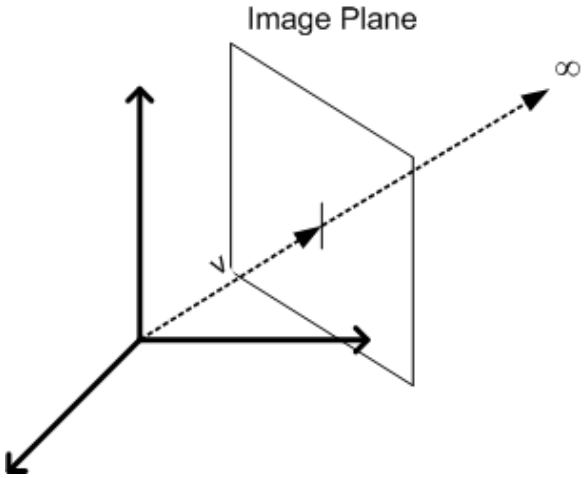



Figure 3.4: The vector $\overline{v}$ is pointing to a single pixel on the image plane. The surface that color came from lies on $\overline{v}$.

has the form:

$$Z = \begin{bmatrix} z_{11} & z_{12} & \dots \\ z_{21} & z_{22} & \dots \\ \vdots & \vdots & \ddots \end{bmatrix}$$

The first step is to translate $(z, \phi, \theta)$ data into $(x, y, z)$ data.

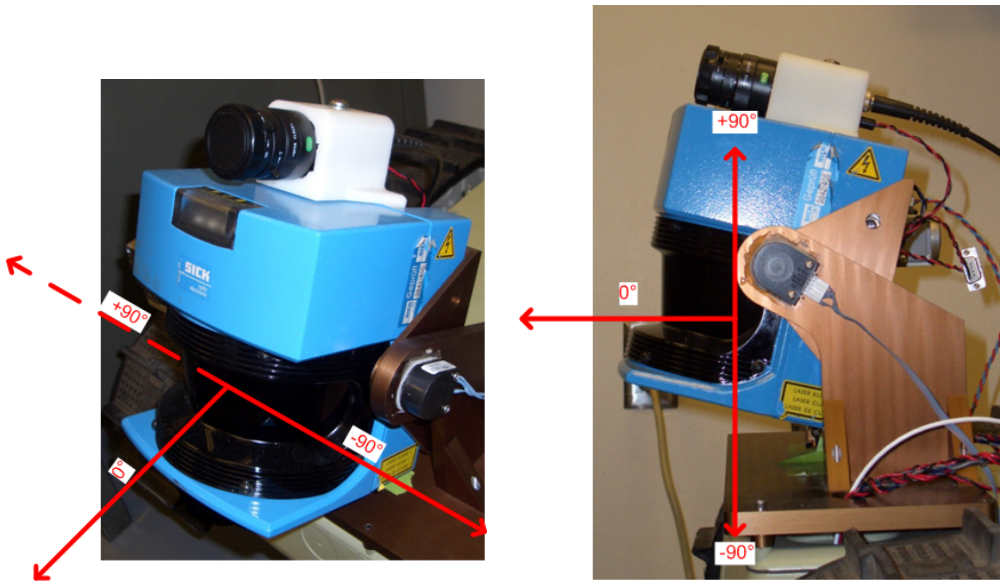
$$\begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 & 0 \\ -\sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} \cos(\phi) & \sin(\phi) & 0 & 0 \\ -\sin(\phi) & \cos(\phi) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} z_{ij} \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

**Coordinate Translation**

Now that the data is in $x, y, z$ form, it is time to translate the range data into the camera's frame by first passing it through the robot's reference frame and then into the camera's. Refer to 2.1 for the homogeneous three-dimensional translation matrix.

## 3.2 Achieving Higher Resolution

Once the range information and image are in the same frame, they may be fused into a colorized range image. However, the laser scanner has lower spacial resolution than the camera. If each range point was back-projected onto the image plane to retrieve a color, the higher resolution of the image would be lost. To maintain resolution, each pixel is projected out onto the range data. This could have been accomplished by simply determining which range points vector is closest to each image pixels vector using dot products. Because of the higher image resolution, multiple pixels would be associated with a single range point.




(a) (b)

Figure 3.5: The laser scanner with scan (a) and tilt (b) angles defined.

This results in colorized range images with serious anomalies. To reduce these anomalies the laser range data is first segmented into a fine-grained object model. With this model, any given range point considers its neighbors to only be those adjacent scan points who have been segmented as belonging to the same object. Using only these neighbors, a least-squares plane fit is run. The normal vector associated with the range point is saved. Image pixels are then projected out from the camera frame and receive a range where they intersect a plane defined by the nearest range point and its normal vector. With this method, pixels that fall on the boundary between two objects do not project to a range where no object resides. This typically occurs in areas where the range data was blocked by an object located closer to the laser scanner. Objects produce a "shadow" of occlusion similar to the shadow they would produce if the a source of light replaced the laser scanner. No range points reside in this shadow.

### 3.2.1 EM Pre-segmentation

The first step in range-color fusion is to estimate which range points belong to the same object. The Expectation Maximization algorithm, using three-dimensional (x, y, z) Gaussian densities, has been shown to segment range data into classes where all members of a class are from the same real object. Here, EM is used to pre-segment the range data into a somewhat arbitrary number of classes (64). The number of classes was chosen to be greater than the actual number of objects in an environment. This number was determined experimentally after examining range data from multiple scenes. Over-segmentation is preferable to under-segmentation because with over-segmentation, classes do not span more than one real object.

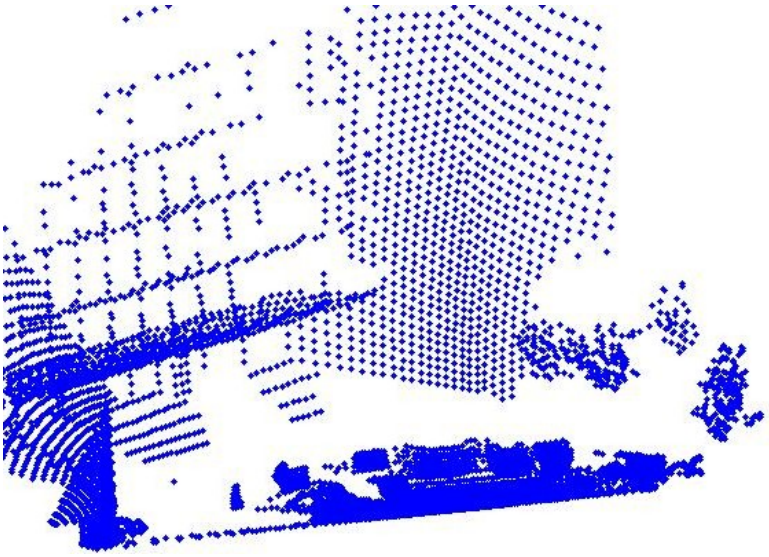


Figure 3.6: The range data after being translated.

Problems occur in the under-segmented case when a class spans multiple objects, as the neighboring points used to compute the plane of projection come from different objects. This causes anomalous pixel projections that adversely affect the later segmentation process on the fused data.

### 3.2.2 Norm Vector Calculation

Once each range point has been segmented into a class, a polygon for each class can be determined. Keeping the range data in two-dimensional form as discussed in 3.1.2 makes it easy to determine a range point's neighbors. A point and its neighbors (maximum of eight) that are in the same class are used to calculate a least-squares plane fit. This will create several planes per class. Planes can be described by the equation $0 = ax + by + cz + d$, where $(a, b, c)$ describes the plane's normal vector and $d$ is defined as $-ax - by - cz$. First, the plane equation must be re-arranged.

$$\begin{aligned} 0 &= ax + by + cz + d \\ 0 &= \frac{a}{d}x + \frac{b}{d}y + \frac{c}{d}z + 1 \\ -1 &= \frac{a}{d}x + \frac{b}{d}y + \frac{c}{d}z \end{aligned}$$

Re-define $a' = a/d, b' = b/d, c' = c/d$

$$-1 = a'x + b'y + c'z$$

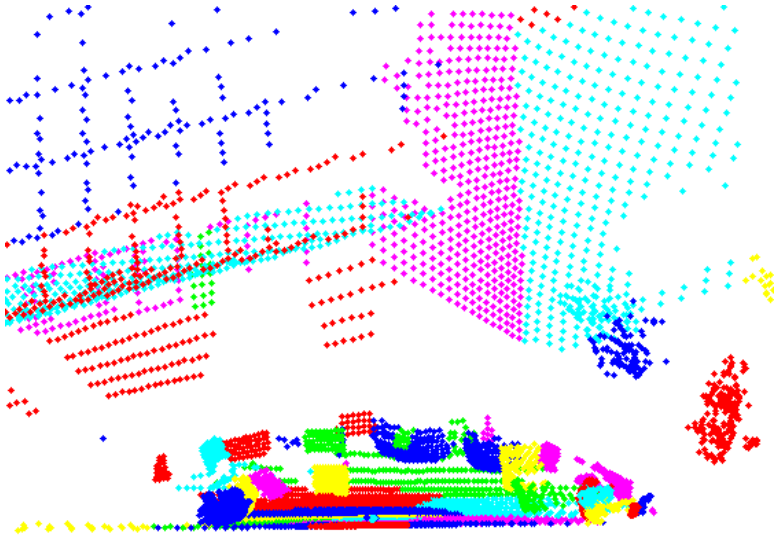


Figure 3.7: Color-coded range data. Each color represent a class after the EM pre-segmentation step.

This equation is now in a form suitable to find a least-squares solution for $(a, b, c)$ given multiple $(x_i, y_i, z_i)$

$$B = A \cdot \underline{x}$$

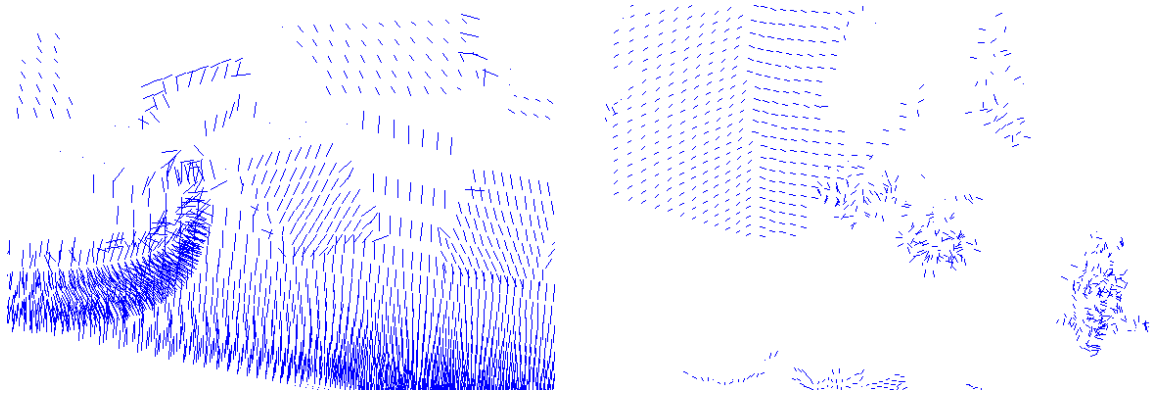
$$\begin{bmatrix} -1 \\ \vdots \\ \vdots \end{bmatrix} = \begin{bmatrix} x_1 & y_1 & z_1 \\ \vdots & \vdots & \vdots \\ x_m & y_m & z_m \end{bmatrix} \cdot \begin{bmatrix} a' \\ b' \\ c' \end{bmatrix}$$

$$A^{-1} \cdot B = \underline{x}$$

The important part of the normal vector is its direction, not its length. Because of this, each normal vector is finally normalized to a unit length. Notice the behavior of normal vectors on different objects in Figure 3.8. Planar objects contain uniform unit vectors. The normals for potted plant base located in the left of 3.8a all fan away from the center. The normals for trees seen on the right of 3.8b have more a random nature.

## 3.3 Data Fusion

Every pixel in the undistorted image defines a vector due to the geometry of the image plane and focal length. Two parameters from the camera calibration come into play into making this calculation. First, the focal length $[fc]_{2x1}$ is different in the y-axis than it is in the x-axis. Secondly, the principle point $[cc]_{2x1}$, the point with the least distortion, is not directly in the center of the image. To account for these two effects, each pixel is first moved



(a) Normal vectors for the floor, the chairs, and the potted plants.

(b) Normal vectors for a wall and some trees

Figure 3.8: A planar fit considering only adjacent scan points segmented within the same class provides a normal vector for each range point.

by the amount of the principal point then scaled by the focal lengths.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x_i - cc(1) \\ cc(2) - y_i \end{bmatrix}$$

This moves the image so the principle point, and not the image center, is at $(0, 0)$. Next, the image vector is scaled by the focal lengths.

$$\begin{bmatrix} x'' \\ y'' \end{bmatrix} = \begin{bmatrix} x'_i * \frac{fc(1)+fc(2)}{2} * fc(1) \\ y'_i * \frac{fc(1)+fc(2)}{2} * fc(2) \end{bmatrix}$$

At this step in the process, the rectified image pixels are ready to be projected out onto the pre-segmented range data. First, each pixel vector is normalized, then its intersection with the planes defined by the laser range data is computed as follows. Consider $p_0 = (x_0, y_0, z_0)$ as the pixel vector of interest, where $|p_0| = 1$. Two points on a plane must satisfy $\underline{n} * (p - p_1) = 0$ where $p$ and $p_1$ are both points on the same plane and $\underline{n}$ is the plane's normal vector. In our case let $p_1 = K * p_0$ where $K$ is a constant defining the length along $p_0$ where it intersects the plane. Therefore, for each image pixel, we would like to find $K$ such that: $\underline{n} * (p - K * p_0) = 0$. Solving for $K$:

$$\begin{aligned} \underline{n} \cdot (p_0 - K \cdot p) &= 0 \\ \underline{n} \cdot p_0 - \underline{n} \cdot K \cdot p &= 0 \\ \underline{n} \cdot K \cdot p_0 &= \underline{n} \cdot p \\ K &= \frac{\underline{n} \cdot p}{\underline{n} \cdot p_0} \\ K &= \frac{ax + by + cz}{ax_0 + by_0 + cz_0} \end{aligned}$$

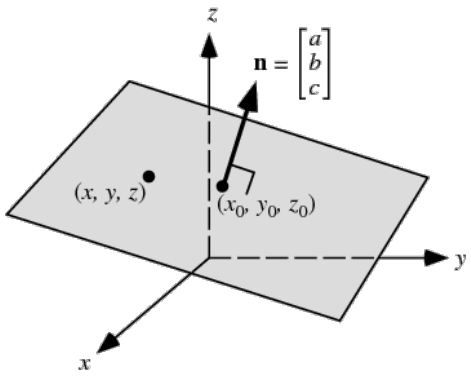



Figure 3.9: Image taken from http://mathworld.wolfram.com/Plane.html

So, $K$ is simply the plane's normal multiplied by a point we know is on the plane divided by the normal multiplied by an image pixel vector. All that is left is to determine which plane the pixel came from. As discussed previously, it is assumed the image pixel came from the same plane as the closest range point. One way to determine the closest point is to take the dot product of the image pixel vector with all the range point vectors.

$$a \cdot b = |a||b| \cos(\theta)$$

Given $|a|$ and $|b|$ are both 1,

$$|a||b| \cos(\theta) = \cos(\theta)$$

As you can see, the dot product of two unit vectors is simple the cosine of the angle between them. The closest range point will produce the dot product that is closest to one. Therefore, we wish to find:

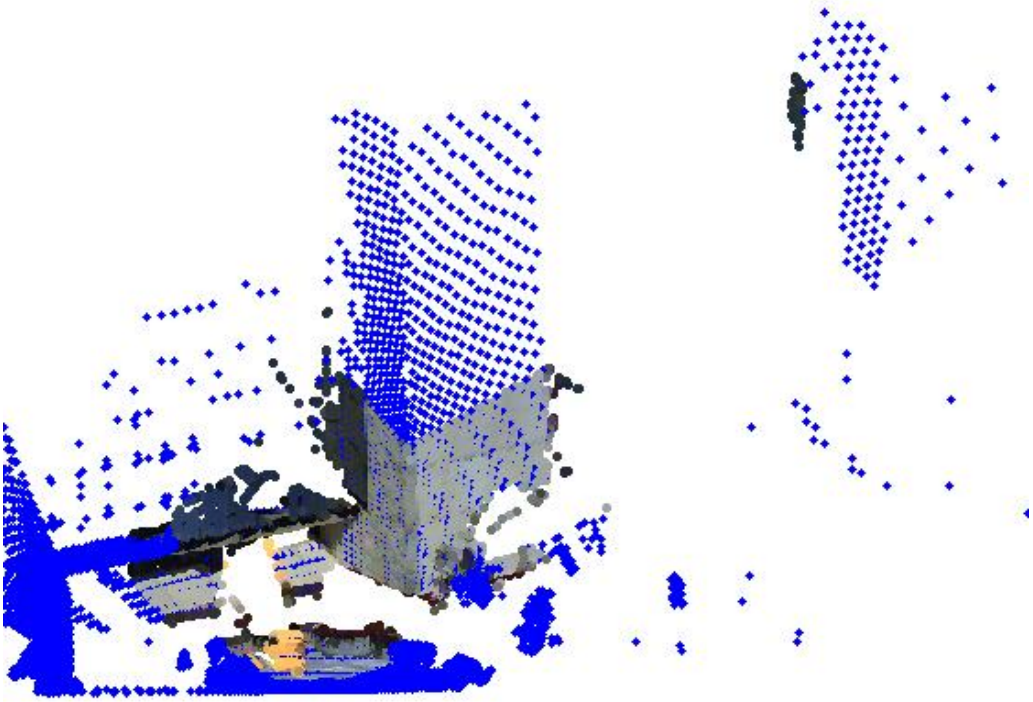
$$\max \left( \begin{bmatrix} x_0 & y_0 & z_0 \end{bmatrix} \cdot \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} \right)$$

Once a range vector is found whose dot product is maximized, $K$, can be calculated and finally project $p_0$.

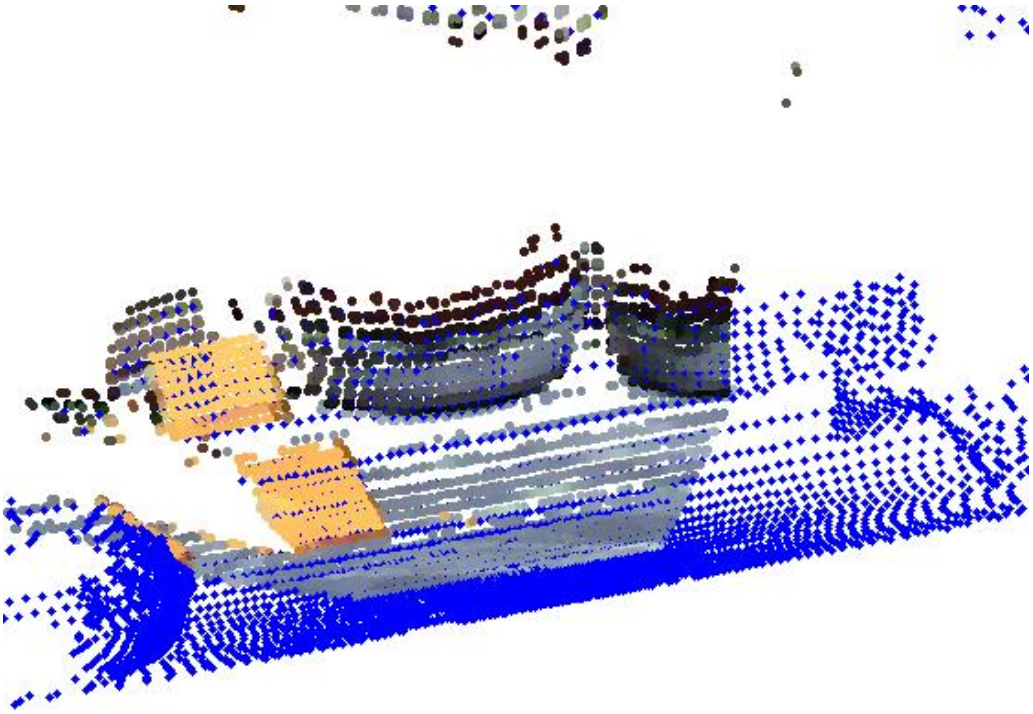
## 3.4 Output File

The final step is to save the data into a file. Only points with an $r, g, b$ and $x, y, z$ data are saved. The structure of the file includes all 6 parameters. The author has written a MATLAB function, `readFile.m`, to automate the reading and matrix creation of an opened file. The file organization goes as follows,

$$\begin{bmatrix} x_1 & y_1 & z_1 & r_1 & g_1 & b_1 \\ x_2 & y_2 & z_2 & r_2 & g_2 & b_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & z_n & r_n & g_n & b_n \end{bmatrix}$$



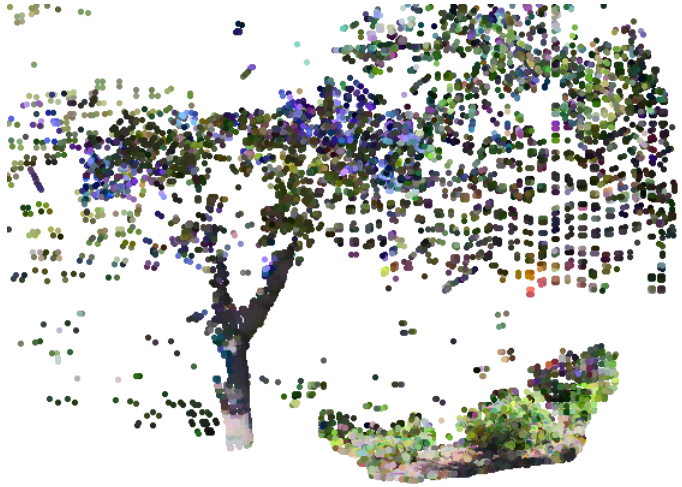
(a) Color Data Superimposed on Top of Range Data



(b) A Closer Look

Figure 3.10: Data from the camera and the laser scanner are fused by projecting each pixel into the scene. The original range data is shown in blue. Note the field of view for the camera is significantly less than the field of view for the laser range scanner.

(a) A Geometric Centerpiece Outside Amongst Trees



(b) All Bushes and Trees

(c) A Person Sitting In An Indoor Environment

Figure 3.11: Fused range-color data for some other environments.

# Chapter 4

# Object Segmentation

With the colorized range image in hand, the next goal is to segment the data into objects using information of both color and range. This work assumes no prior knowledge of the environment. Also, the exact number and type of objects present in the scene are both unknown. Existing methods for extracting objects from data such as this are often tailored to one specific environment like inside buildings or around city blocks. The goal of this research has been to develop a method suitable for generic objects both indoors and outdoors. The object model must apply to rigid geometric bodies as well as more loosely defined objects. To accomplish this, a six-dimensional Gaussian density is used as a universal object model. The parameters of the object model are exactly the mean and covariances of the distribution for the x,y and z coordinates and the r,g and b color information.

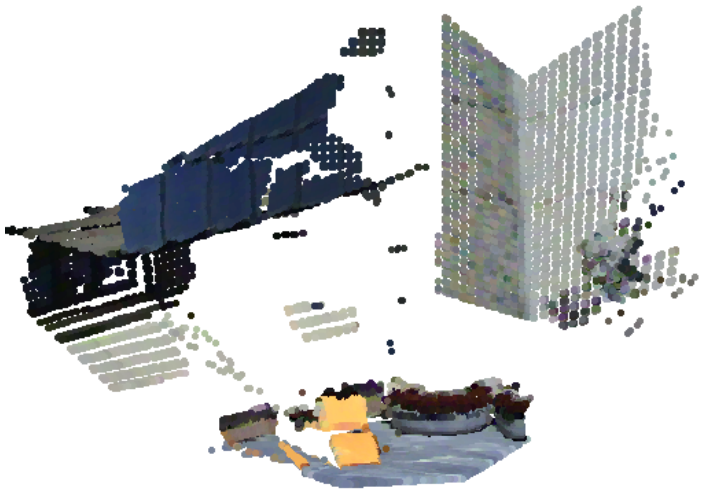
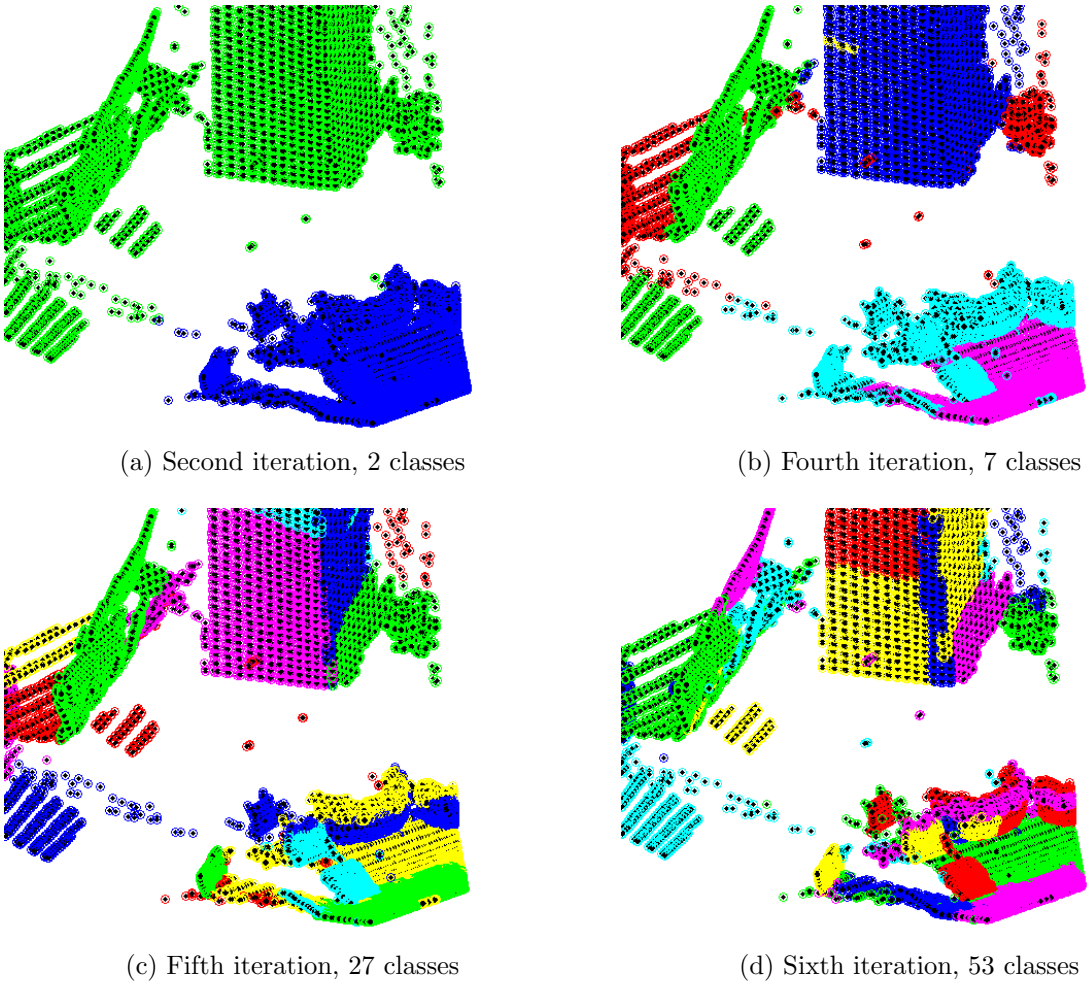


Figure 4.1: A colorized range scan of an outdoor scene containing both planar objects such as walls and floor and non-planar objects such as the orange chairs, the tree, and the potted plants.

## 4.1 Repetitive EM

The first step in locating an unknown number of objects in an unknown environment is to successively process subsets of colorized range data, dividing each subset into two classes. The goal is not to determine how many real objects are in a dataset, but rather to generate a collection of classes where members of a class do not come from two different real objects. Again, it is better to over-segment a data set. The repetitive EM step is divided into three phases: splitting, reclassification, and pruning.

Beginning with a single class containing all the data, the first splitting step returns two classes. Each of these two classes is then split again and again until a fine grained segmentation is achieved. Similar to the earlier range-only segmentation phase, the goal is to sufficiently subdivide the data into classes where no one class contains data from more than one physical object. Each class is parameterized by a six-dimensional Gaussian density function (x,y,z,r,g,b).



(a) Second iteration, 2 classes

(b) Fourth iteration, 7 classes

(c) Fifth iteration, 27 classes

(d) Sixth iteration, 53 classes

Figure 4.2: Different iterations of the repetitive EM step.

The resulting fine grained segmentation is crude. Data from a single object often falls into multiple classes. Therefore, the entire data set is reprocessed. This time all at once, with an EM implementation considering all the object models calculated in the splitting phase. Unlike the splitting phase, the reclassification phase does not increase the number of classes, but allows the data from an object to be reclassified into a single model. Objects that were artificially split may be joined. Some classes grow while some classes shrink. The reclassification phase also modifies the distribution models. As can be observed in the results in Figure 4.2, although some physical objects are represented with multiple classes, no class contains multiple objects. The next section will further refine this segmentation by fitting simple geometric models to each class and merging classes with similar models.

After each splitting phase, the total number of classes would double. However, after some iterations, some classes contain only a small number of points. These classes are considered poorly defined and removed in the pruning phase. Each point from the removed class is then associated with the class of its nearest neighbor. Because of this pruning phase, the total number of classes after each iteration is not double the previous. The pruning phase is also run after each reclassification phase for the same reason. For this thesis, the repetitive EM step was looped six times. Figure 4.2 displays the result of some of the iterations for the repetitive EM step. The plotting algorithm plots by cycling through six different colors. All the green points, for example, are not part of the same class.

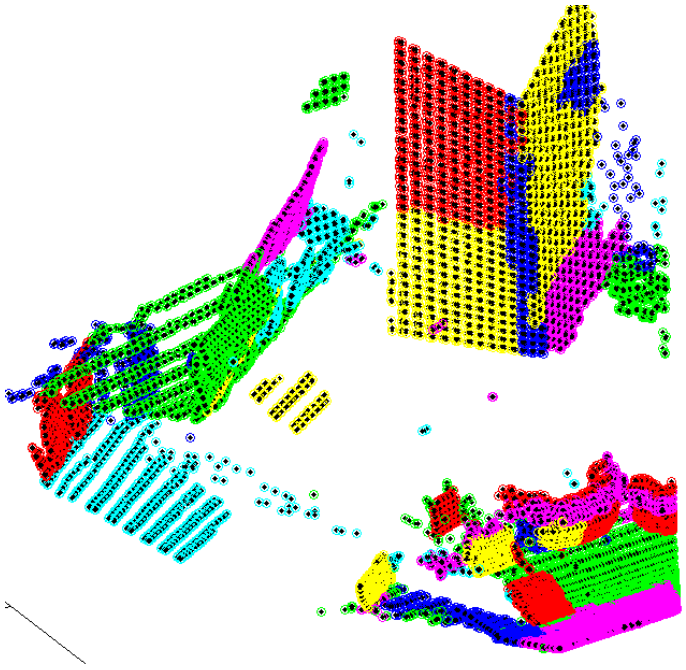


Figure 4.3: Groups of points are segmented into color-coded classes using several iterations of the Expectation Maximization algorithm. Note that while an objects may have several classes, no single class spans multiple objects.

## 4.2 Planar Extraction

Many scenes contain objects that are man-made. Typically, these objects are characterized well by planar objects. Locating and segmenting planes from a data set would identify a variety of man-made objects. The planar extraction step attempts to locate planes in the fused data using the segmented classes. To achieve this, a least-squares plane fit is performed for each class. If the variance of the residuals is small, that class is deemed a plane. In addition, similar planes are merged, reducing the total number of classes. Figure 4.2d displays all the planar classes found in the scene. Note that the purple wall is now a single plane.

The repetitive EM step returns a class number for each data point. As you can see in Fig. 4.3, while some objects contain multiple classes, there are no classes that contain multiple objects. In the planar extraction step, all classes that contain points from the same plane will be merged into a single class. The purpose of this step is two fold. One, planar objects usually "attract" a large number of classes. By extracting these classes and merging them, the total number of classes for the dataset has been reduced while planar objects have been detected without prior knowledge of their existence, location, or number. Two, if planes can be accurately extracted, it may be possible to search for other models in future development of this algorithm. To achieve this, every class will be run into a least-squares plane fit algorithm. The MATLAB function `lsplane.m` returns the centroid of the data $x0$, direction cosines of the normal of the best-fit plane $\Gamma$, a vector containing a residual error per point $\Psi$, and the norm of the residual errors $normd$[14]. Examining the variance of the residual errors proved to be an effective criterion for points that actually lie on the same plane. Any class of points with a residual variance less than $2cm^2$ was a plane.

Planes will be merged based on their distance $D$ from the origin. This distance is calculated using the direction cosines $l$, $m$, and $n$ located inside the vector $\Gamma$. These are the cosines of the angles the normal vector $\overline{n}$ makes with the $x$, $y$, and $z$ axes respectively.

$$l = \cos(\alpha) = \frac{\overline{n} * \widehat{x}}{|\overline{n}|}$$
$$m = \cos(\beta) = \frac{\overline{n} * \widehat{y}}{|\overline{n}|}$$
$$n = \cos(\gamma) = \frac{\overline{n} * \widehat{z}}{|\overline{n}|}$$

The normal vector $\overline{n}$ is of unit length, therefore $|\overline{n}| = 1$ which simplifies the above equations to

$$l = \overline{n} * \widehat{x}$$
$$m = \overline{n} * \widehat{y}$$
$$n = \overline{n} * \widehat{z}$$

In other words, as long as $\overline{n}$ is of unit length, its direction cosines are equivalent to its normal vector.

Recall the equation for a plane: $0 = ax + by + cz + d$ where $(a, b, c)$ is the surface's normal vector. In the function `lsplane.m`, $\Gamma$ returns direction cosines, which is equivalent to returning the surface's normal vector $(a, b, c)$. The variable $d$ can now be calculated by $d = -ax - by - cz$ using the plane's centroid for $(x, y, z)$ located in $x0$. The distance $D$ the plane is from the origin is calculated by

$$D = \frac{d}{\sqrt{a^2 + b^2 + c^2}}$$

Because $(a, b, c)$ is of unit length, $D$ is equivalent to $d$. Planes with difference in $d$ of $10cm$ are merged together. Figure 4.4 displays the final classes that were determined to be planar.

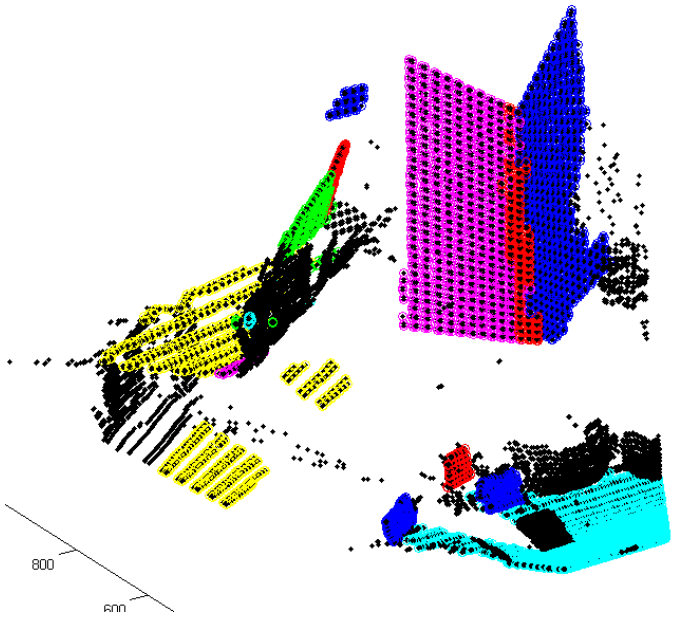
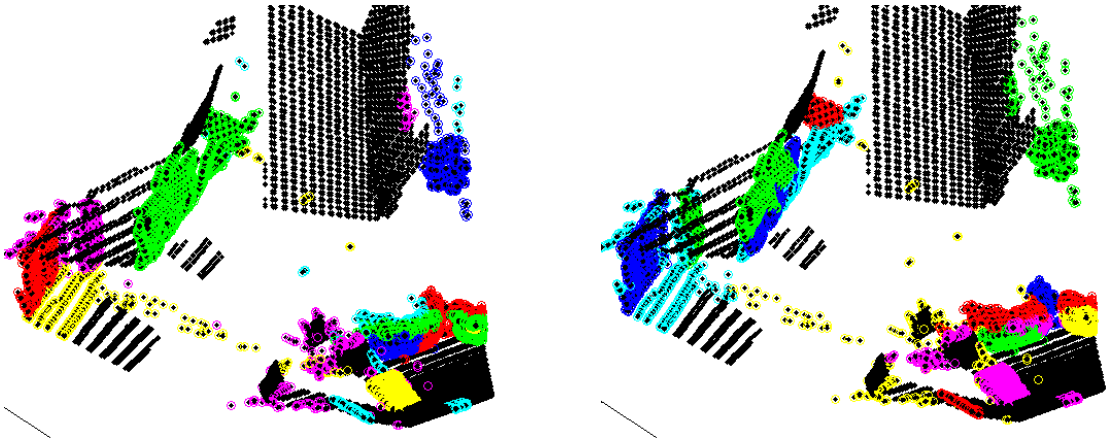



Figure 4.4: Color-coded planar regions after planar merging. Points in black were determined not to belong to any planar object.

## 4.3 Second-Run EM

By clustering planar classes, many classes are merged. With the planes removed from the dataset, it may be possible for the remaining classes to adjust more accurately to the real objects that remain. For the second-run EM step, points that were not determined to be planar are run through EM again. This time EM maximizes the remaining number of classes. For example, if there were initially 53 classes and the planar clustering step clustered 20 of those classes into 5 planes, the second-run EM step would maximize 33 classes. Even after removing the data points associate with the planes, the remaining classes are still maximized. Instead, the second-run EM step does not use the classes' parameters for its initialization step. Instead, the author uses the initialization discussed in 2.3.2. This step is only run if the planar extraction step removed several planes from the dataset. Otherwise, this step does not provide any additional segmentation.



(a) (b)

Figure 4.5: The non-planar classes before (a) and after (b) the second-run EM step.

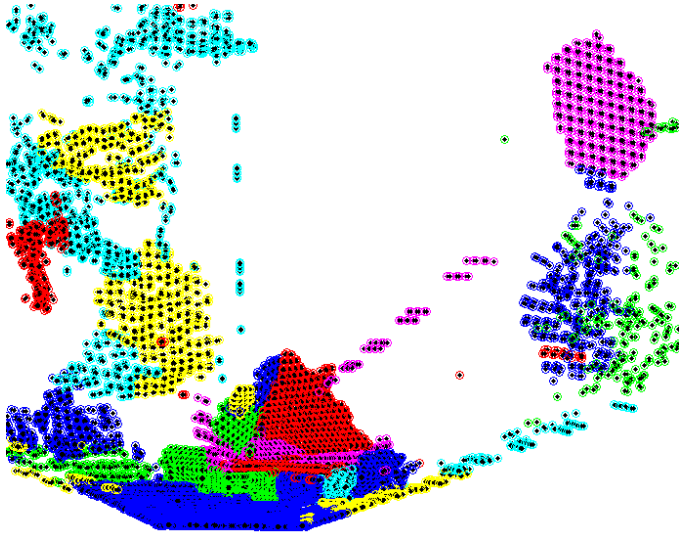
## 4.4 Analysis

The algorithm described in this thesis was tested on several different environments: one purely indoors, one with only trees and bushes, and two with a combination of geometric objects and natural objects. In all, certain key features were accurately segmented. Figure 4.6 shows the classes for these other environments.

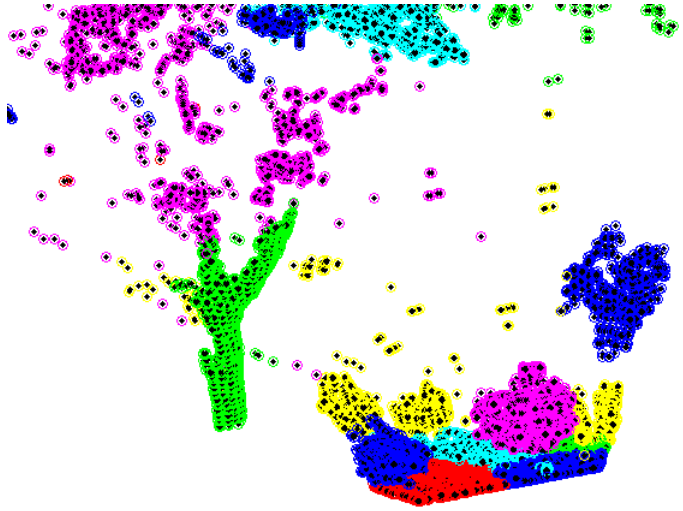
The planar extraction algorithm did remove all tightly defines planes from the environments as you can see in Figure 4.4. The planar merging portion correctly identified all three classes located on the ground of Figure 4.3 to belong to the same plane. Also, the classes on the rectangular pillar (4.3) were reduced from seven classes to three. Several planes on the other portions of the building were also correctly identified and merged.

With the planes removed from the data set, real objects that were still left to be identified were: portions of the orange chairs, two potted plants, one tree, and some parts to the building. After the second-run EM step, many of these objects reside in a single class. In Figure 4.5b, the tree was successfully segmented as was the bottom of the chair and the small portion of a second chair. Interestingly, points that were mis-projected were grouped together. The potted plants' bases were correctly segmented, while the flowers in both were grouped into one class.

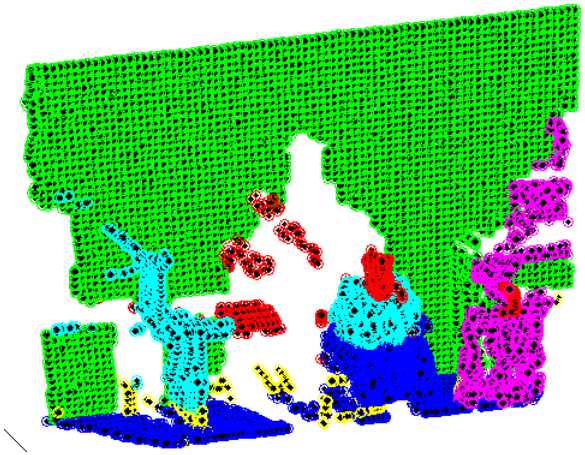
It may be noted in Figures 4.5b and 4.4 that there are anomalies. Some are explained by the fact that the building in the background is primarily made of glass, which reflected most of the range data. This led to a large gaps in the range information. The merging algorithm attempted to still project color information into these spaces, with limited success. The pixel projects do not correctly match the real surface (glass) that they came from. Where there should only be one plane for the glass surface, there are several smaller planes, all with different normal vectors. Other anomalies occur because the points in the classes were either too small or did not provide enough information of an accurate least-squares plane fit.



(a) A Geometric Centerpiece Outside Amongst Trees



(b) All Bushes and Trees



(c) A Person Sitting In An Indoor Environment

Figure 4.6: The color-coded final results from three other environments.

# Chapter 5

# Conclusion and Future Work

This thesis presented a method to fuse laser range data and color information gathered from a camera utilizing the Expectation Maximization algorithm and normal vectors. The resulting dataset contained r,g,b and x,y,z information for data points. These data points were then segmented using universal object segmentation and six-degree Gaussian densities. This procedure also utilized the Expectation Maximization algorithm. Preliminary results, like those in Figure 4.6, look promising. Highly planar objects are easily extracted and clustered from the Gaussian densities. It may also be possible to segment other shapes, such as cylinders or spheres, out of the Gaussian densities. The clustering and segmentation also occurred without prior knowledge of the type of objects in the dataset, or their number. This keeps the algorithm applicable to indoor, outside, and mixed environments.

The author notes several areas for future research. One area that plays an important role in segmentation is the fusing of range and color data that creates the dataset to be segmented. The author's algorithm to perform this fusion was used because it utilized the resolution of the camera, but did not require significant processing time. The future goal for range-color fusion should not be higher resolution, but more accurate pixel projections. Many pixels from the image were either not projected, due to insufficient data, or projected incorrectly. These incorrect projections distort objects, sometimes blurring the space in between objects. Another way to improve range-color fusion would be to scan an environment from several locations and mesh these scans together. Having more information about an object should make them easier to segment.

For the initial idea of the six-degree Gaussian density, range and color values are handled by EM with equal weighting. The range measurements had units of centimeters while the color information was represented rgb values. Clearly the units are not the same and should possibly be treated differently by EM. Many researchers[8] argue that the hsv color space more closely relates colors as interpreted by humans and is less susceptible to changes in illumination.

While most concrete floors and man-made walls were correctly segmented in the planar extraction step, some were not. These surfaces and the ground in outdoor environments usually failed the tight variance test. One option could be to perform RANSAC instead of a least-squares shape fit. RANSAC should be able to handle noisy surfaces better.

# Bibliography


[1] S. Pu and G. Vosselman, Automatic extraction of building features from terrestrial laser scanning.

[2] G. Vosselman, B. Gorte, G. Sithold, and T. Rabbani, Recognising structure in laser scanner point clouds, in *Proceedings of Conference on Laser Scanners for Forest and Landscape Assessment and Instruments*, 2004.

[3] S. Biswas, G. Aggarwal, and R. Chellappa, Invariant geometric representation of 3d point clouds for registration and matching, in *2006 IEEE International Coference on Image Processing*, 2006.

[4] T. Abmayr et al., Realistic 3d reconstruction - combining laserscan data with rgb color information.

[5] V. Sequeira and J. ao G.M. Gonçalves, 3d reality modelling: Photo-realistic 3d models of real world scenes, in *Proceedings of the IEEE First Internation Symposium on 3D Data Processing Visualization and Transmisstion*, 2002.

[6] L. S. Nyland, Capturing dense environmental range information with a panning, scanning laser rangefinder, 1998.

[7] P. Dias, V. Sequeira, F. Vaz, and J. ao G.M. Gonçalves, Registration and fusion of intensity and range data for 3d modelling of real world scenes, in *Proceedings of the 4th IEEE International Conference on Recent Advances in Digital Imaginand and Modeling*, 2003.

[8] Y. Pang, Q. Huang, X. Quan, J. Zheng, and X. Wu, Fast object location and tracing using two ccd cameraws and laser range finder, in *Proceedings of the IEEE International Conference on Information Acquisition*, 2005.

[9] S. Thrun et al., IEEE Transactions on Robotics and Automation **20** (2004).

[10] P. Biber, H. Andreasson, T. Duckett, and A. Schilling, 3d modeling of indoor environments by a mobile robot with a laser scanner and panoramic camera, in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2004.

[11] N. Ueda, R. Nakano, Z. Ghahramani, and G. E. Hinton, Split and merge em algorithm for improving gaussian mixture density estimates, in *Neural Networks for Signal Processing VIII, 1998. Proceedings of the 1998 IEEE Signal Processing Society Workshop*, 1998.

[12] K. Strobl, W. Sepp, S. Fuchs, C. Paredes, and K. Arbter.

[13] C. A. Bouman, Cluster: An unsupervised algorithm for modeling Gaussian mixtures, 1997.

[14] I. Smith.

[15] G. Bertolini and S. Ramat, Identification and recognition of objects in color stereo images using a hierachial som, in *Fourth Canadian Conference on Computer and Robot Vision*, 2007.

[16] M. Waltherm, P. Steinhaus, and R. Dillmann, *A Foveal 3D Laser Scanner Integrating Texture Into Range Data*, volume 0, 2006.