

# **Environmentally Friendly Robot: Protecting Natural Reserves against Random Littering**

**Honors in Research Program Thesis**

**Advisor: Professor William Sethares**

**Electrical and Computer Engineering Department**

**University of Wisconsin Madison, WI 53706**

**BY DENG, XIANG**

**May 2015**

**© Copyright by Deng, Xiang 2015**

**All Rights Reserved**

# Acknowledgements

I would like to thank my advisor, Professor Sethares, for his advice, wisdom and tremendous help over the past two years. I can still recall the moment when I was sitting in the first row in his Image Processing class two years ago; over the past four semesters, he provided me with the freedom of working on my own but constantly helped me via emails and instilled me with a lot of confidence to go through the obstacles. His advice, not only sheds lights on my technical questions, but guides me to battle against my procrastination and various difficult problems that have been encountered during the project or will be seen in the long run.

I want to express my gratitude to Professor Yu-Hen Hu for his teaching and advice on ‘features’. I would like to thank my art teacher Allison Welch, for encouraging me to see, explore and listen to the world and bring my soulfulness and creativity to my pursue in robotics.

To all my friends, I want to thank you for sharing the moments with me and standing by my side over the years.

This research received the funding from the Hilldale Research Fellowship. Their support gave me much confidence at the beginning of the work. This research utilized the following open source tools: OpenCV, Mobile Robot Programming Toolkit (MRPT), and Point Cloud Library (PCL).

谢谢我的父母，从我有生以来一直无私的付出。Especially, I want to thank my parents, for their endless love and support throughout my life.

# Presentations

The findings of this research have been presented in the following events:

- Honors Thesis Seminar, UW Madison, May 1<sup>st</sup>, 2015, EHall 4610, UW-Madison
- Engineering EXPO 2015 (also a second place winner), April 16-18<sup>th</sup>, 2015, ECB, UW-Madison <http://engineeringexpo.wisc.edu/exhibits/>
- Undergraduate Research Symposium, April 16<sup>th</sup>, 2015, Northwoods B, UW-Madison [https://ugradsymposium.wisc.edu/view\\_abstract.asp?id=4926](https://ugradsymposium.wisc.edu/view_abstract.asp?id=4926)
- Sustainability Lunch Hour, April 10<sup>th</sup>, 2015, Science Hall 360, UW-Madison <http://sustainability.wisc.edu/hour/>

# Abstract

With large increases in tourism, it is difficult to maintain parks and preserves in their natural state due to random littering. The purpose of this research project is to develop an automated robot that is capable of maneuvering in a dynamic and unstructured wilderness environment to collect data about the form and distribution of litter. The robot is intended to provide critical information about the commonly contaminated areas and to help tourist sites develop a more efficient method of collecting trash. The distribution patterns will allow for cleaning crews to quickly find trash in common areas, thus reducing labor costs and helping preserve the natural beauty of the park.

## Table of Contents

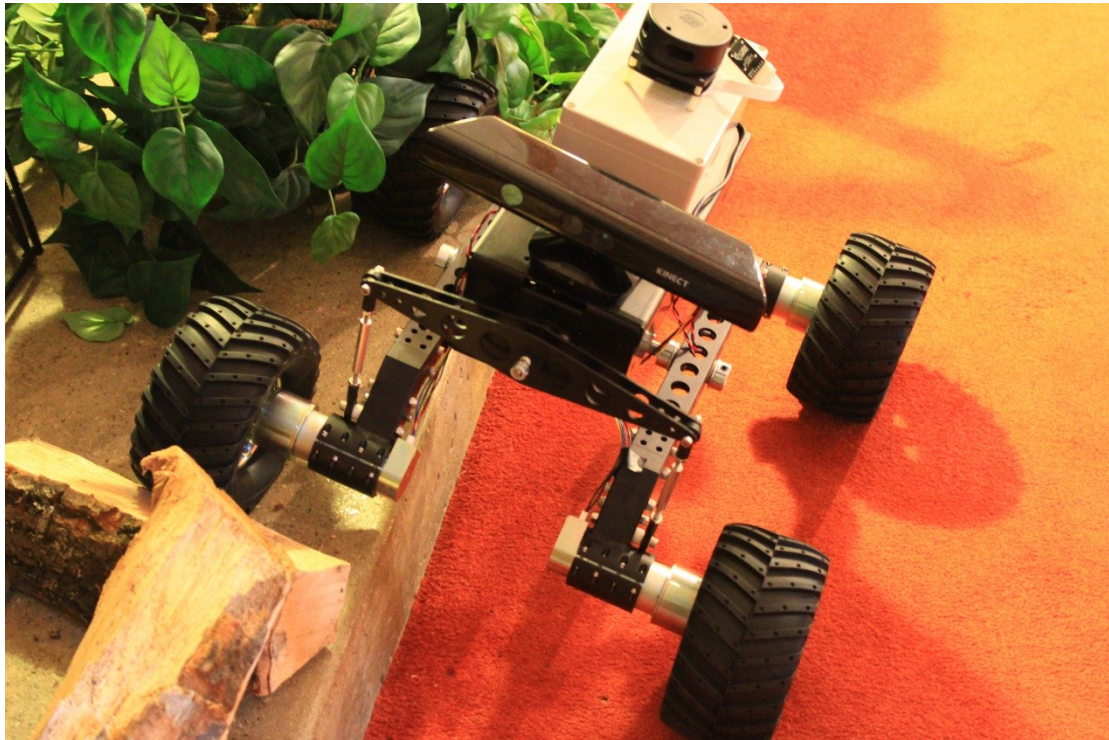
Environmentally Friendly Robot: Protecting Natural Reserves against Random Littering .....	1
Acknowledgements.....	2
Presentations .....	3
Abstract .....	4
Chapter1. Introduction.....	6
Chapter2. Simultaneous Localization and Mapping.....	8
2.1 Background: Representing the world using Occupancy Grid Maps.....	8
2.2 Goal of Study .....	11
2.3 2D SLAM .....	12
2.3.1 The Iterative Closest Point Process .....	12
2.3.2 Matching, Alignment and Accumulation of Errors .....	13
2.3.3 Proposed Solution: Two Layers of Mapping.....	15
2.4 RGB-D SLAM .....	17
2.4.1 Proposed RGB-D SLAM algorithm .....	17
2.4.2 Rejection.....	18
2.5 Hybrid 2D-RGBD SLAM .....	27
2.5.1 Proposed Algorithm.....	28
Chapter3. Image Segmentation and Object Classification.....	32
3.1 Segmentation with Texture Learning and energy minimization .....	32
3.1.1 Literature reviews on the most relevant: a.Clustering with Connected Components Labeling .....	32
3.1.2 Literature reviews on the most relevant: b.Energy Optimization.....	33
3.1.3 Literature reviews on the most relevant: c.Texture.....	34
3.1.4 Proposed RGBD Segmentation with One-Shot Texture Learning and Refinement	35
3.2Object Classification (In progress).....	38
3.2.1 Studies on Texture classification .....	38
3.3 Proposed Training and Classification method (In progress).....	43
Future Works: On the way to self-learning and knowledge transmission across generations .....	44
References.....	45
Appendix .....	49

# Chapter 1. Introduction

Since my childhood, I have witnessed the increasing random littering and pollution in nature reserves, along with the acceleration of tourism in my home country of China. However, the currently adopted solution for this problem, which entails cleaners going around the nature reserves for the purpose of searching and collecting garbage, however the purpose of searching and collecting garbage is highly inefficient. Moreover, travelling in different developing and developed countries as I grew up, I found that the random litter disposal is a common issue related to education, human resource management, economy, and welfare. In order to achieve effective waste management strategies, understanding the litter distribution pattern and interpreting visitor's littering behavior is essential. Artificial Intelligence (AI), which employs machine learning and probabilistic reasoning theories, can be used to learn the distribution of litters in nature reserves and generalize the pattern, thereby shedding light on this problem.

Driven by this dream, I chose AI and Robotics as my research areas when I started undergraduate education in UW-Madison in 2012. Since January 2014, under the supervision of Professor William A. Sethares, I have been working on my four-semester independent autonomous robot research project, "Environmentally Friendly Robot: Protecting Natural Reserves against Random Littering".

This thesis contains two major chapters that detail my study and major accomplishments in the following aspects: Simultaneous Localization and Mapping (SLAM), Image Segmentation and Object classification. The final proposed Hybrid 2D-RGBD SLAM, which combines both 2D and RGBD measures and uses each when it works best, aims to facilitate robots path planning and autonomous navigation in more dynamic, unstructured unknown environment. The Image Segmentation with One-Shot Texture Learning and Refinement Process, computes the geometric and color features, performs texture analysis and energy minimization for optimal solution. It also allows robot to learn about the texture of new environment in order to achieve a more intelligent, efficient segmentation system. In terms of efficiency, in general cases according to testing, it only requires one iteration of learning and refinement. While the study in object classification is still in progress, the current learning patterns sheds light on the classification of garbage among the natural environment—it turns out that it may not just be a problem of distinguishing "garbage vs. non-garbage", but also "natural vs. non-natural".



The robot Dali (or Caramel), with laser scanner (top), Kinect RGBD sensor (front), and Intel Core i5-4250U Processor

Considering the robot's learning potential in mapping, image segmentation and classification as well as handling uncertainties in unknown environment, in the future, I wish to transform the current project into others that may contribute to education, water scarcity and many more...I also wish that the developments in robot, can be introduced as new factors to many sustainability aspects such as the long term environmental equilibrium. When thinking about the future, I always wonder: Can robots awaken people's mind to better protect the earth and ourselves? I am looking forward to see the answer through the rest of my life.

# Chapter2. Simultaneous Localization and Mapping

## 2.1 Background: Representing the world using Occupancy Grid Maps

Recovering spatial representations of the surroundings is essential for autonomous robot's path planning and decision making. Best-known map representations include occupancy grid map [1], point cloud [2] and landmarks [3]. Since a robot can only understand its environment indirectly from sensors data, recovering the spatial representation of the surroundings must deal with the uncertainty and noises exists in the sensor data.

The occupancy grid map partition the space into cells, while each cell stores a probabilistic estimate of whether the cell is occupied or not. Among the three major map representations, the occupancy grid map has the following advantages:

1. Best estimate of the surroundings. The occupancy grid map can be incrementally updated by fusing multiple sensory data, eg. Laser range scans, given the pose estimates of the sensor corresponding to each sensory data. The fusion between the current observation and previous observation can be done via Bayesian approach. By iteratively accumulating the sensor data samples and updating the occupancy cell, the occupancy cells give more accurate probabilistic estimate of cell occupancy
2. Reduction of the small variance exists in sensory data. By correctly setting the resolution of the grid map for a given application, spatially close enough data that refer to a same target can be fused into a single cell instead of being treated as two obstacles.
3. The occupancy grid map, essentially can be treated as a binary map and directly used for path planning methods such as A\* search.

For clarification, below is a review of how Occupancy Grid Map works.

Definitions:

$P(Oxy)$ : the probably of a cell being occupied for a cell with index  $(x,y)$ .

$r_t$ : observation (distances to obstacles) at time  $t$

$x_t$ : the pose of the sensor/robot at time  $t$

$I_t$ : the information at time  $t$ ,  $r_t \wedge x_t$

$J_t$ : all information until time  $t$

$P(Oxy | I_t)$ : the likelihood of the cell  $(x, y)$  being occupied, given the sensor pose  $x_t$  and



distance to obstacle  $r_t$

Initially, without any sensory information, we donate  $P(O_{xy}) = P(\neg O_{xy}) = 0.5$

According to Bayes's theorem [1],

$$P(O_{xy}|I_t \wedge J_{t-1}) = \frac{P(I_t|O_{xy} \wedge J_{t-1}) * P(O_{xy}|J_{t-1})}{P(I_t|O_{xy} \wedge J_{t-1}) * P(O_{xy}|J_{t-1}) + P(I_t|\neg O_{xy} \wedge J_{t-1}) * P(\neg O_{xy}|J_{t-1})} \quad (2.1)$$

That is, given the new information and old information, the likelihood of cell (x,y) being occupied. This is the incremental update step of the cell occupancy by incorporating the current knowledge and previous observations.

According to [4], if we make a strong assumption between  $I_t$  and  $I_{t-1}$ , then the odd of the above will be  $\frac{P(O_{xy}|I_t \wedge J_{t-1})}{P(\neg O_{xy}|I_t \wedge J_{t-1})} = \frac{P(O_{xy}|I_t)}{P(\neg O_{xy}|I_t)} * \frac{P(O_{xy}|J_{t-1})}{P(\neg O_{xy}|J_{t-1})} * \frac{P(\neg O_{xy})}{P(O_{xy})}$  (2.2)

Alternatively, it can be written as  $Odds(O_{xy}|I_t \wedge J_{t-1}) = \frac{Odds(O_{xy}|I_t) * Odds(O_{xy}|J_{t-1})}{Odds(O_{xy})}$  (2.3)

Note the range of the odds is from 0 to infinity, if we take the log of equation (2.1), then the range will be from negative infinity to infinity:

$$\text{Log}(O_{xy}|I_t \wedge J_{t-1}) = \text{Log}(O_{xy}|I_t) + \text{Log}(O_{xy}|J_{t-1}) - \text{Log}(O_{xy}) \quad (2.4)$$

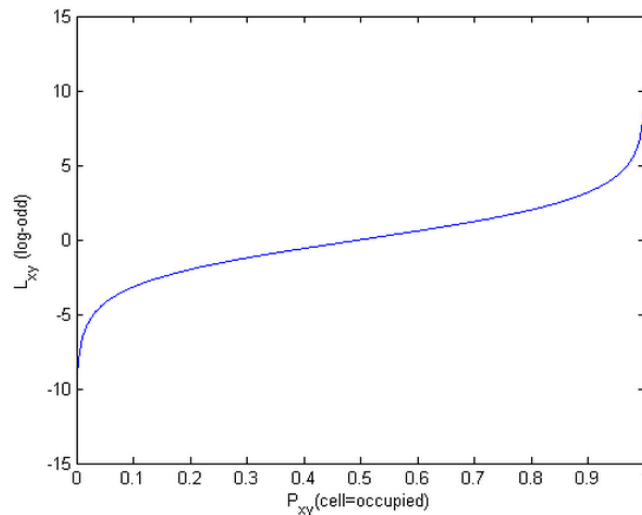


Figure 2.1.1 Log-odds vs. Cell Occupancy.

Source: [http://www.mrpt.org/tutorials/programming/maps-for-localization-slam-map-building/occupancy\\_grids/](http://www.mrpt.org/tutorials/programming/maps-for-localization-slam-map-building/occupancy_grids/)

In common cases, for a given pose and laser range data, we can assume the probability density curve of  $P(O_{xy}|rt)$  will look like: [5]

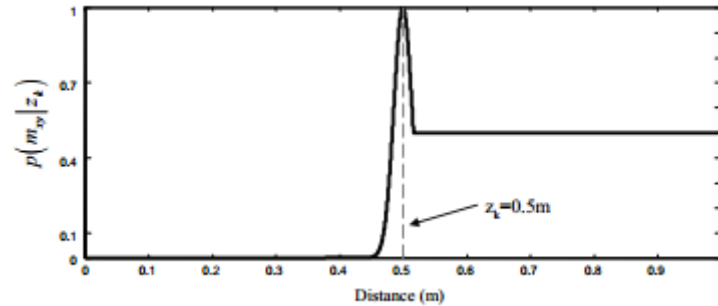


Figure 2.1  $P(O_{xy})$  given the range to an obstacle is 0.5m [5].

This means, given the range to an obstacle, if the distance between a cell and the robot is very close to the measured distance, the cell is probably occupied; if a cell is further away from the observed point, we keep the default value 0.5 for the cell occupancy, as it is still unseen.

The above summarizes the process of incrementally updating and estimating cells occupancies. By accumulating samples, we will estimate the density of  $P(O_{xy}|t - 1)$  and the average cell density  $P(O_{xy})$ ; for a given pose,  $P(O_{xy}|It)$  can be assumed as shown in Figure 2.1. For a given cell occupancy, if new observation confirms with the previous estimate, the occupancy likelihood will increase and vice versa. Therefore, this model can be regarded as a simple learning model; over time, it takes samples and gives a better estimation of cells occupancies as well as the spatial representation of the global. Its applications in this study and potential issues will be discussed in the following sections.

## 2.2 Goal of Study

Simultaneous Localization and Mapping (SLAM), is a technique that allows robots to update a map of its surrounding and localize it while navigating in unknown environment. For path planning, action planning and probabilistic reasoning, it provides one fundamental basis. The key idea of mapping is to constantly match and align the current sensory data to the previous observations; by aligning the new sensory data to the existing map, the robot will also be able to determine its current pose.

The ultimate goal of this research project is to create an autonomous robot exploring in unstructured, dynamic nature reserves. The following cases are mostly common while the robot navigates in dynamic environments (suppose the sensory data comes from laser rangefinder):

Situation A. A sudden change in robot's pose (tilt or roll) causes the sudden change in the laser scan pattern. In this case, the matching ratio between the new laser scan and existing map may potentially be low.

Situation B. A sudden change in the structure of the local environment. Once the robot enters the critical zone, a change in surroundings may also bring about a bad correlation between the current and previous observations; however, if this change can be detected, the robot can start building a new map while long term path planning can be based on a multiple local maps.

Situation C. No sudden change in the environment structure or the robot's pose ever happened; however, the changes in pose and the structure of the environment accumulate over time.

In terms of the situations above, two key questions must be answered: How to determine the pose and possibly register new sensory data into existing map when there is a low matching ratio between the current and previous observations due to the sudden pose change or environment change? How to represent the spatial characteristics of the surroundings for long term and short term path planning ( eg. Should we should 2D or 3D models or both)?

The proposed solution in this chapter is a feedback system incorporates both 3D and 2D matching methods and take advantages of both measures. The proposed method also fuses all sensory information into 2D global map for long term planning; local sensory data will be for more accurate, short term path planning. Before introducing the proposed hybrid system, studies and developments in 2D and 3D SLAM will be first discussed separately in detail.

## 2.3 2D SLAM

### 2.3.1 The Iterative Closest Point Process

Iterative Closest Point (ICP) is one widely-used method that can be used to find the best alignment between two point clouds. For this study in 2D SLAM, it plays a key role in matching and aligning new sensory data with previous observations. The key concepts include: for each point in the source point cloud, find its corresponding point in the target cloud; based on the corresponding pairs, estimate the transformation matrix between two point clouds; iteratively repeat the previous steps until the algorithm converge.

For clarification, below is the algorithm [6]:

Definitions:

Source Point cloud:  $A = \{a_1, a_2, \dots, a_n\}$ , Target Point cloud:  $B = \{b_1, b_2, \dots, b_n\}$

Estimated Transformation Matrix:  $T$

Initial guess in transformation:  $T_0$

Weight of pair  $i$ :  $w_i$

Max Euclidean distance between the corresponding pairs:  $\text{dist\_threshold}$

Algorithm 2.3.1:

Let  $T = T_0$

while *not converged*

  for  $i$ : 1 to  $n$

$m_i = \text{FindClosestPointInB}(T * a_i)$

    if  $\|m_i - T * a_i\| > \text{dist\_threshold}$

$w_i = 0$ ;

    else

$w_i = 1$ ;

  end

  According to all corresponding pairs with  $w_i = 1$ , estimate the overall transformation  $T_i$  using a mean squared error cost function;

$T = T_i$ ;

  if *converge or no improvement*

    break;

end

Note that the ICP will converge to local minimum, that is, as long as no further improvement can be made, it will stop, even though the alignment is incorrect. Therefore, initial estimate of the transformation matrix is critical for the success of ICP.

Also choosing `dist_threshold` brings about a tradeoff between convergence and accuracy [6]---if `dist_threshold` is too low, the algorithm will stop between it actually converge; if `dist_threshold` is too high, incorrect correspondences will also contribute to the estimation of transformation matrix.

In terms of computation efficiency, a KD-tree can be used for the search of nearest neighbors between two point clouds [7].

### 2.3.2 Matching, Alignment and Accumulation of Errors

So far, we have talked about the concepts of occupancy grid map and ICP for point clouds alignment. This section examines the possibility of using incorporating both methods for 2D SLAM, and potential issues.

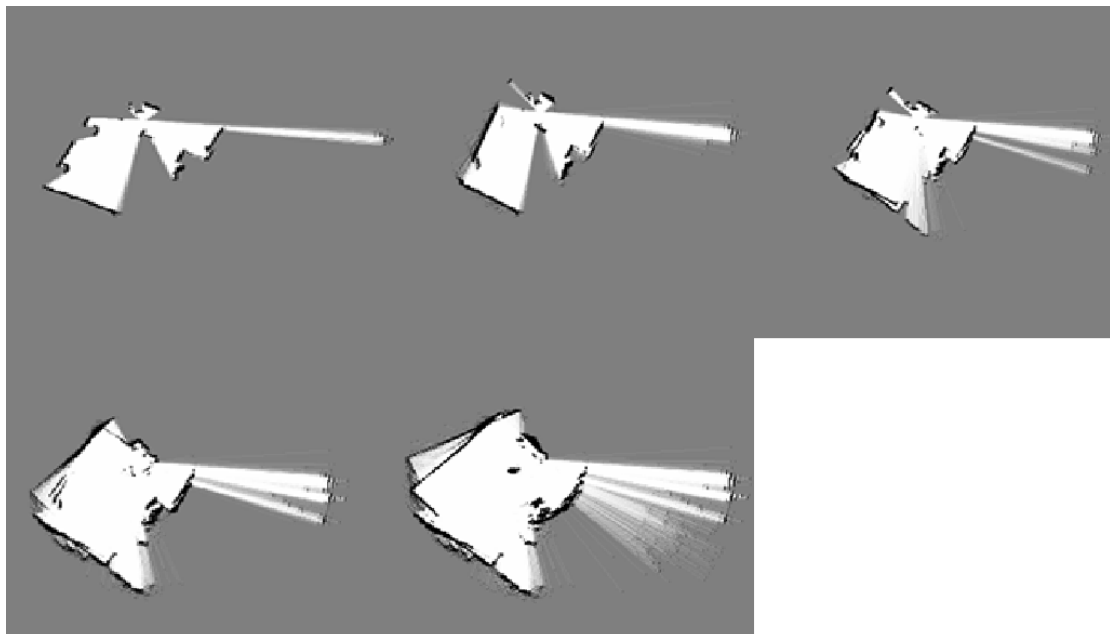


Figure 2.3.1 An illustration of the ICP based registration process; the sensor is static; however, after registration of multiple frames of laser range scan, the system diverge and end up with a wrong map building.

Figure 2.3.1 demonstrates a ICP based registration process: the laser rangefinder is static and repeatedly scans the surroundings in a room setting; new laser scan is aligned with the built map using ICP approach (the existing map is resampled as points cloud) and then registered into the global grid map. Registration error accumulates; over time new sensory data faces multiple possible alignments, while ICP causes random local convergences, leading to the system to eventually diverge.

Suppose both the sensory data and the alignment are perfect, then the situation described in

Figure 2.3.1 may not occur. However, in reality, due to uncertainty, noises in sensory data, there is almost no perfect alignment between new and previous laser scan. In other words, the transformation between subsequent scans may not be perfectly linear or rigid.



Figure 2.3.2, a zoomed in version of the second map in Figure 2.3.1

Figure 2.3.2 is a zoomed in version of the second map in Figure 2.3.1. Most data from new laser scan agree with previous scans; however, due to noises or early convergence of ICP, there is no perfect overlap between the new and old observations. For next frame of data, it faces multiple possible alignments. Accumulation of errors explains why the system will diverge.

In 2.1, we mentioned that the occupancy grid map, by learning from more samples, can give a better estimation of the surroundings. Unfortunately, in this case, the growth of errors is dominating the errors rejection. The occupancy grid map is not best used in this approach. Next section will be the discussion of the proposed solution as well as the examination of its effectiveness.

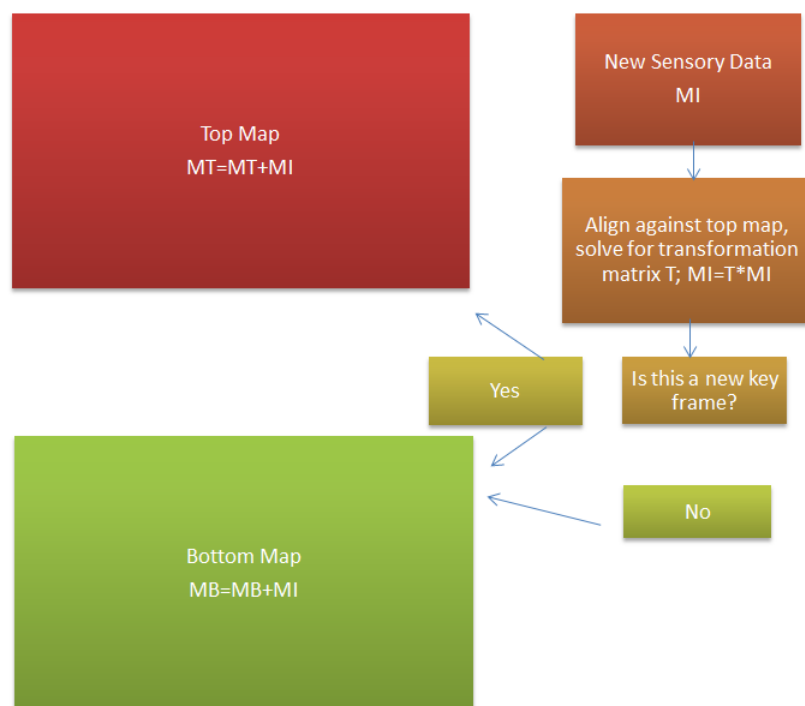
### 2.3.3 Proposed Solution: Two Layers of Mapping

Admittedly, in many mobile robot applications, people can avoid the problems mentioned in the previous section by only registering key frames. In other words, only register new sensory data when the change in robot's pose indicates a new environment is probably detected. Issues with this approach is that noises or moving objects won't be flushed out or correctly updated in the global map, while the occupancy map is not learning from more samples in order to give a better representation of the world.

We wonder if we can achieve a system that is able to:

1. Constantly learn from samples in real time and give a best estimate of the surroundings;
2. Reduce or eliminate the accumulation of errors while registering more samples.

Below is the diagram showing the proposed solution:



All new sensory data will be matched against the top map; depending on whether it can be considered as a new key frame, it will be transformed and registered in the bottom map or both. Over time, the bottom map will be an averaged version of the surroundings, temporal changes in the surroundings such as noises or moving objects will be reflected in this map but flushed out over time. Indeed, for the building of the top map which is essentially a map of

key frames, the proposed model makes a strong assumption on the correctness of transformation estimation between a new key frame and the top map. Admittedly, errors may still occur along with the building of the top map, but this problem won't lead system to diverge, and can be solved by methods such as loop closure algorithm [5] [8][9][10].

### Tests

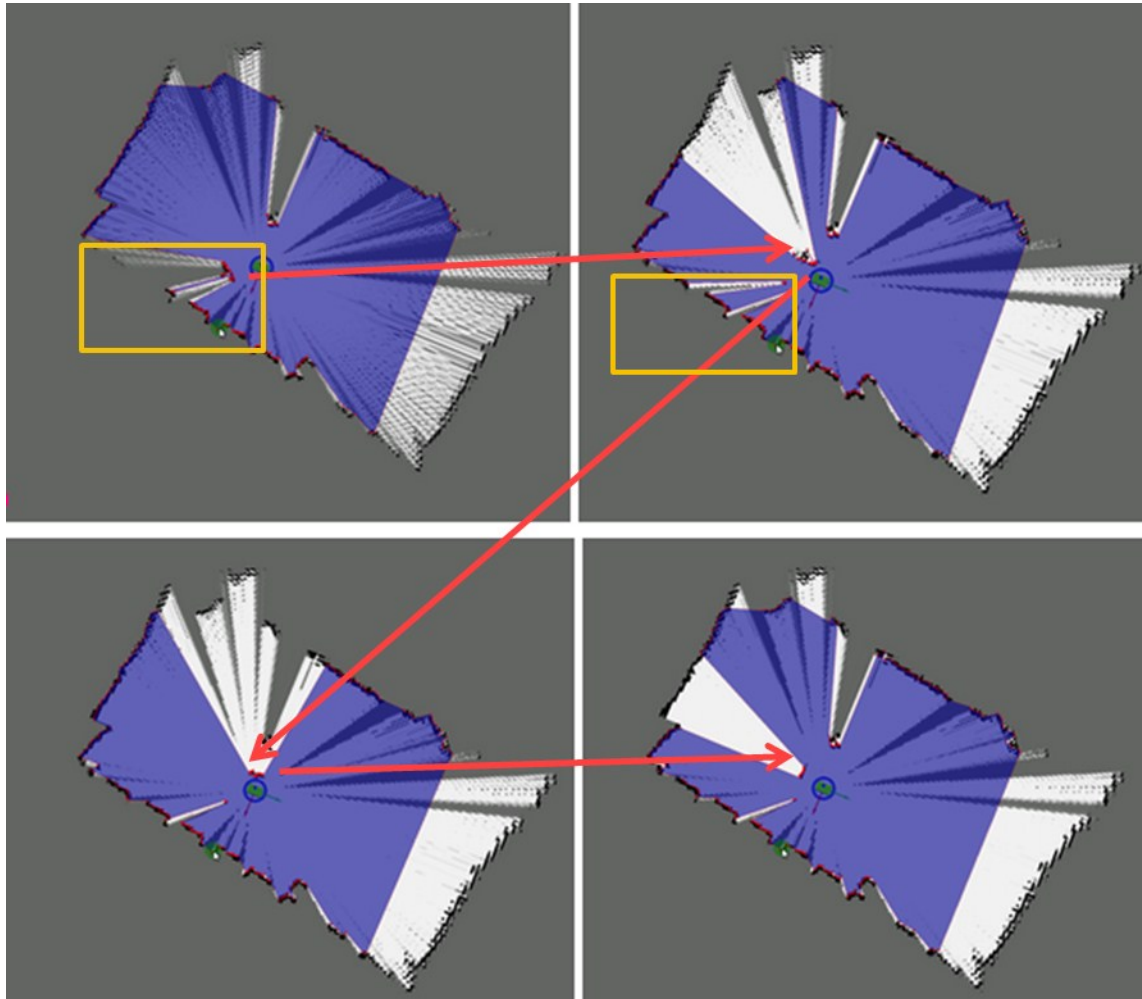


Figure 2.3.3.1 A learning process from multiple observations. While the obstacle is moving, previously blocked area (yellow boxes) has been updated, and the movement of obstacle (red arrow) is constantly updated on the map.



## 2.4 RGB-D SLAM

### Introduction

In contrast to ICP-based 2D registration as discussed in 2.2, alignment of RGB-D images (point clouds) is typically done by first finding the matched image features and estimating the rigid transformation. Without initial guesses, running ICP algorithm for finding the nearest neighbors among the two dataset and repeatedly aligning two clouds can be extremely expensive, and may converge to incorrect local minimum. We use SURF (Speeded Up Robust Features) for detecting matched features between consecutive images [12]. As RGB-D images don't provide full 3D data but depth per pixels, other widely accepted 3D descriptors [13][14][15][16] are studied but not used in this case. Indeed, SURF will give false matches. After SURF matching, we reject bad correspondences using a proposed rejection method. Based on the current knowledge, SVD (Singular Value Decomposition) will then be used to give the best estimate of the rigid transformation. Due to sensory data noises, image distortion or motion blur, rigid transformation between two clouds is not guaranteed. Again, according to the estimated transformation matrix, we run ICP on all matched SURF points, performing non-linear optimization, seeking for the best alignment. We run ICP on all matched SURF points because: 1. Increasing the sampling size to a reasonable amount: the proposed rejection method is a very rigorous step which may not only discard bad correspondences but also exclude correct matches; 2. all points are already associated while ICP will automatically reject those that are not spatially close enough. Admittedly, in case when there is not enough SURF matches, failure may occur. However, in terms of consecutive images, this case rarely occurs and can be handled by sampling new images.

As summarized in [17], three main topics of SLAM include frame alignment, loop closure detection and global optimization. Due to the scope of this thesis, other two topics won't be discussed here while the proposed RGB-D Alignment algorithm will directly contribute to the Hybrid-SLAM in 2.5.

### 2.4.1 Proposed RGB-D SLAM algorithm

Algorithm 2.4.1 RGB-D Alignment:

```
For the given RGB-D images  $Img_1, Img_2$  and their point clouds  $PtCloud_1, PtCloud_2$ , convert
to gray scale images  $G_1, G_2$ ;
 $\{P, Q\} \leftarrow \text{FindCorrespondingPointsWithSimilarSURFFeatures}(G_1, G_2, PtCloud_1, PtCloud_2)$ 
For  $i=1$  to length (P)
    If  $p_i$  or  $q_i$  has undefined xyz features, remove  $p_i$  and  $q_i$  from P and Q
End
Create two empty point clouds, PP and QQ
 $\{PP, QQ\} \leftarrow \text{RejectBadCorrespondences}(P, Q)$  (Algorithm 2.4.2)
```

Estimated Transformation Matrix  $T \leftarrow \text{TransformationEstimationWithSVD}(PP, QQ)$  (2.4.3)

$Q \leftarrow T * Q$

$T \leftarrow T * \text{TransformationEstimationWithICP}(P, Q)$  (Algorithm 2.3.1)

$PtCloud_2 \leftarrow T * PtCloud_1$

## 2.4.2 Rejection

### 2.4.2.1 Proposed Bad Correspondences Rejection Method

Here the proposed rejection method is a special case for matching consecutive RGB-D images, with an assumption that the rotation between neighboring frames won't be significant. It resembles a rigorous voting process and has a linear computation complexity. Other best-known algorithms such as RANSAC may be adopted for more general purposes. However, according to experiments, this proposed method shows its robustness and efficiency! Mathematical proof can be found

Algorithm 2.3.2 Proposed Bad Correspondences Rejection Method

Given matched points  $P = \{p_i\}$ ,  $Q = \{q_i\}$

Create a circular histogram  $H = \{h_i\}$  with  $M$  bins

Create a vector  $V$  of integer vectors with size of  $M$

Let  $d\text{-angle} = 360/M$  or  $2\pi/M$

For  $i=1$  to length  $[Q]$

$$T\text{-angle} = \text{ArcTan}\left[\frac{-p_{iy} + q_{iy}}{c - p_{ix} + q_{ix}}\right]$$

$$\text{Index } I = \frac{-\frac{d\text{-angle}}{2} + T\text{-angle}}{d\text{-angle}}$$

$H[I] ++;$

$V[I].\text{pushback}(i)$

End

Select max  $h_i$  with index  $m$

$PP = \{p_i\}$ ,  $QQ = \{q_i\}$  where  $i$  belongs to  $V[m]$

Return  $PP, QQ$

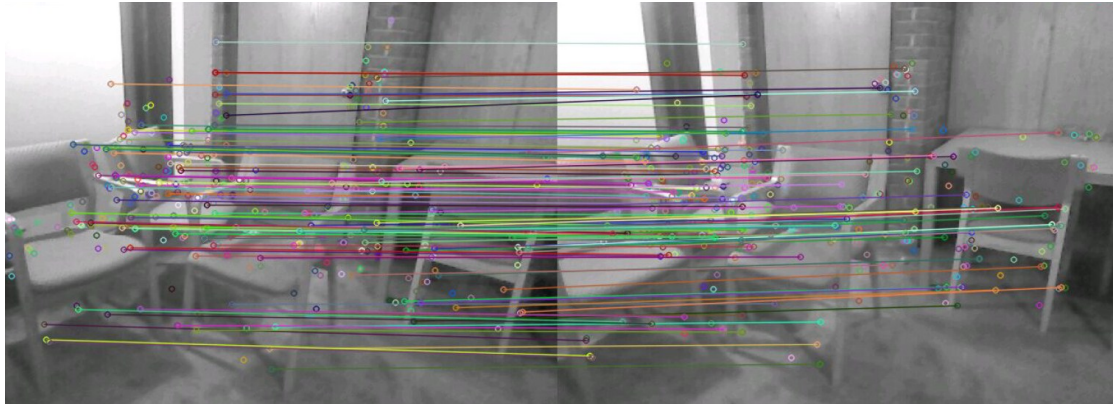


Figure 2.4.2.1 RGB-D images with SURF matches

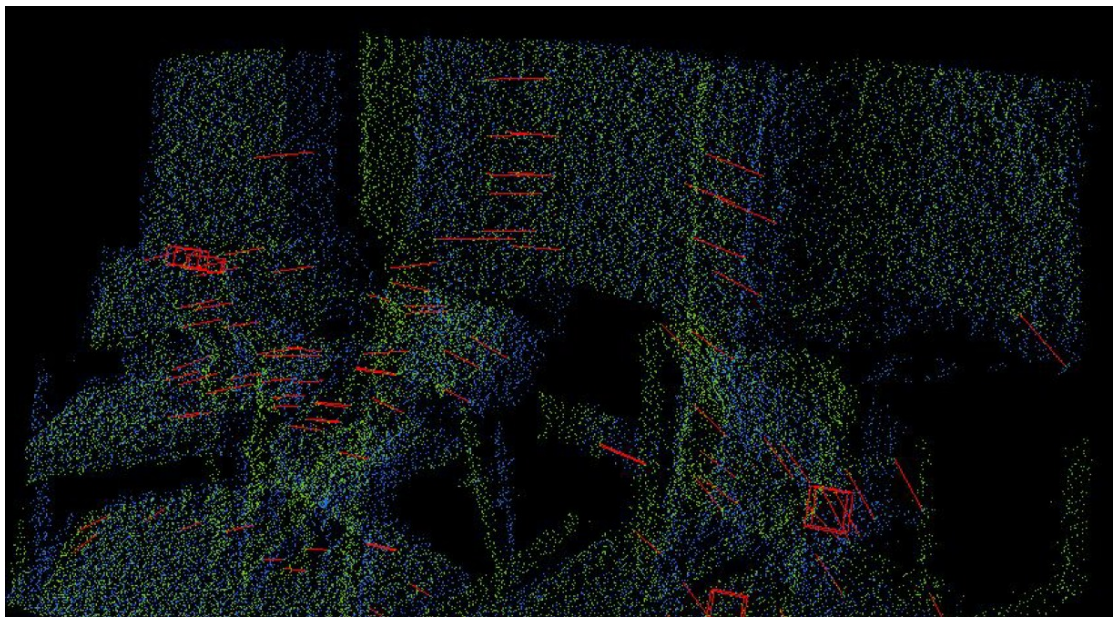


Figure 2.4.2.2 SURF correspondences shown between two point clouds

## 2.4.2.2 Visualization of the Histograms and the corresponding Transformation Matrices

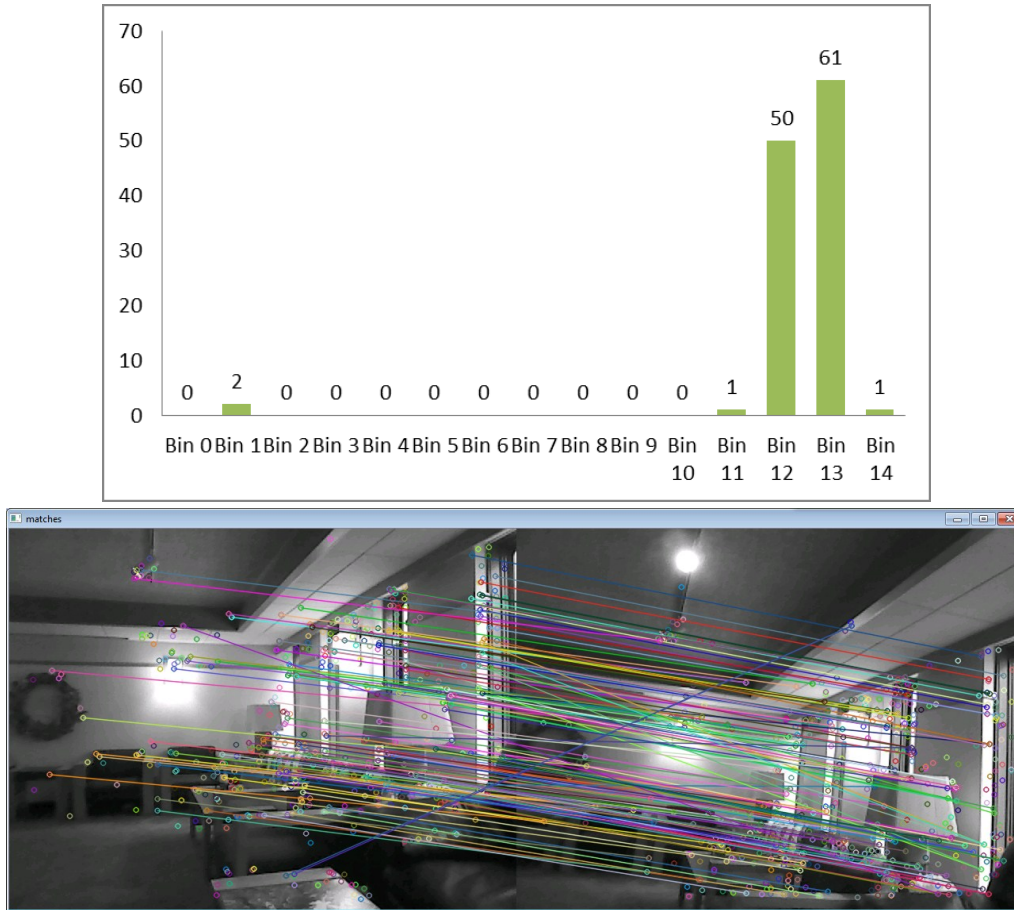


Figure 2.4.2.2.1 Bin Histogram (15 bins), Quantization of angles upon significant tilt

## 2.4.2.3 Mathematical Proof for the Proposed Rejection Algorithm

A strong assumption made here is that the correctly associated 2D points correspond to the 3D points  $p_i$  and  $q_i$  that can be perfectly matched. Since changing the angle of view or moving the camera may distort/blur the 2D perspective, this relationship may not be perfectly true.

In theory, the proposed rejection method is invariant to pitch change, insensitive to yaw change, less insensitive to roll change. Below is the mathematical reasoning and tables:

**Define  $f = \text{ArcTan}\left[\frac{-p_{iy} + q_{iy}}{c - p_{ix} + q_{ix}}\right]$**

For any point  $p_i$ , the matrix representation:

```

pt =

ptx
pty
ptz
1

```

Assume  $q_i$  be the associated point with  $p_i$ , let  $T$  be the transformation matrix from  $p_i$  to  $q_i$ , ideally  $q_i = T * p_i$ .

Let  $x, y, z$  be the translation in x-axis (width), y-axis (height), z-axis (depth),

Let pitch, roll, yaw be the rotation around y-axis, x-axis, z-axis

For the following cases, let  $c=3$ ,  $-1.5 \leq x \leq 1.5$  (width),  $-1 \leq y \leq 1$  (height),  $0.8 < z < 6$  (depth) (normal ranges for Kinect Camera)

The overall 6D transformation matrix will be:

```

[ cos(pitch)*cos(roll) - sin(pitch)*sin(roll)*sin(yaw), -cos(yaw)*sin(roll), cos(roll)*sin(pitch) + cos(pitch)*sin(roll)*sin(yaw), x*cos(roll) - y*sin(roll)]
[ cos(pitch)*sin(roll) + cos(roll)*sin(pitch)*sin(yaw), cos(roll)*cos(yaw), sin(pitch)*sin(roll) - cos(pitch)*cos(roll)*sin(yaw), y*cos(roll) + x*sin(roll)]
[ -cos(yaw)*sin(pitch), sin(yaw), cos(pitch)*cos(yaw), z]
[ 0, 0, 0, 1]

```

**Since there are too many variables to consider, for the following we examine the individual effect of  $x, y, z, \text{yaw}, \text{pitch},$  and  $\text{roll}$  individually.**

### Case1. Effect of pitch change

The transformation Matrix  $T$ :

```

[ cos(pitch), 0, sin(pitch), 0]
[ 0, 1, 0, 0]
[ -sin(pitch), 0, cos(pitch), 0]
[ 0, 0, 0, 1]

```

Then Matrix Representation of  $q_i = T * p_i =$

```

ptx*cos(pitch) + ptz*sin(pitch)
pty
ptz*cos(pitch) - ptx*sin(pitch)
1

```

$$f = -\text{ArcTan}[(pty - ptz) / (c + ptx * \text{Cos}[pitch] + ptz * \text{Sin}[pitch] - ptx)] = 0$$

In this case,  $f$  is invariant to pitch change.

### Case2. Effect of roll change

The transformation matrix:

```

[ cos(roll), -sin(roll), 0, 0]
[ sin(roll),  cos(roll), 0, 0]
[      0,      0, 1, 0]
[      0,      0, 0, 1]

```

Matrix representation of qi:

```

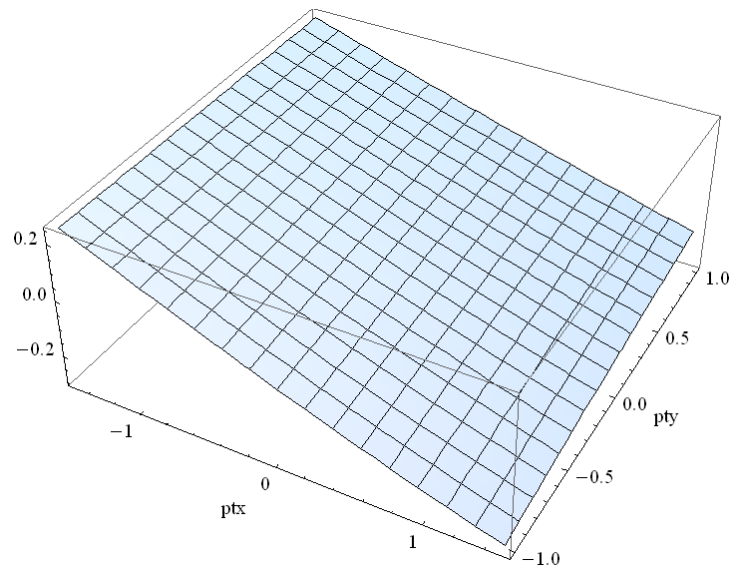
ptx*cos(roll) - pty*sin(roll)
pty*cos(roll) + ptx*sin(roll)
          ptz
          1

```

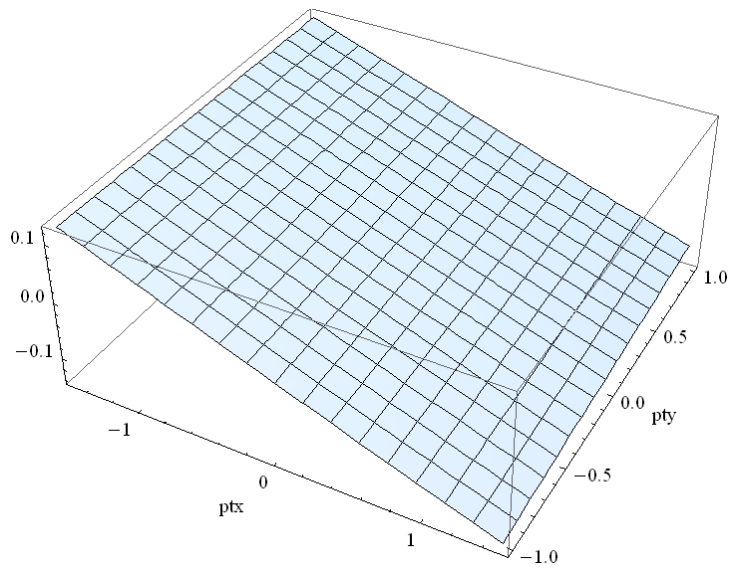
$f$

$$= -\text{ArcTan}[(pty * \text{Cos}[roll] - pty + ptx * \text{Sin}[roll]) / (C + ptx - ptx * \text{Cos}[roll] + pty * \text{Sin}[roll])]$$

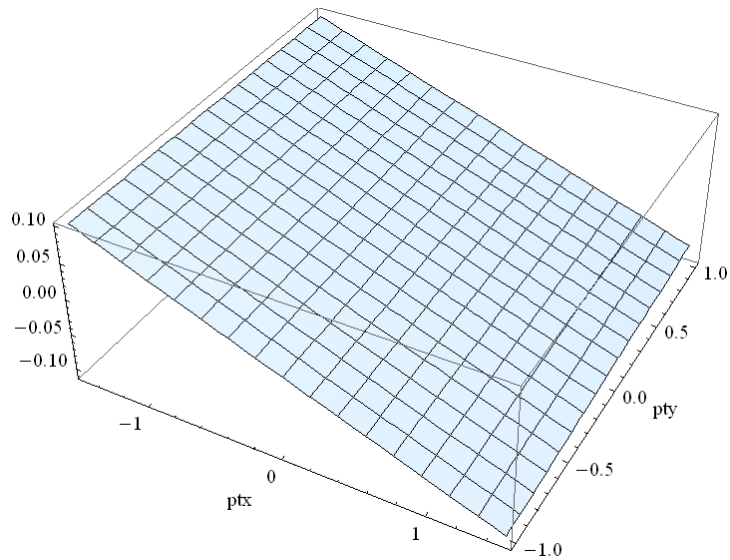
Roll=0.5 radians,  $f_{max} - f_{min} \sim 0.4$  radians



Roll=0.25 radians,  $f_{max} - f_{min} \sim 0.25$  radians



Roll=0.2 radians ,  $f_{max} - f_{min} \sim 0.2$  radians



### Case3. Effect of yaw change

The transformation matrix:

$$\begin{bmatrix}
 1, & 0, & 0, & 0 \\
 0, & \cos(\text{yaw}), & -\sin(\text{yaw}), & 0 \\
 0, & \sin(\text{yaw}), & \cos(\text{yaw}), & 0 \\
 0, & 0, & 0, & 1
 \end{bmatrix}$$

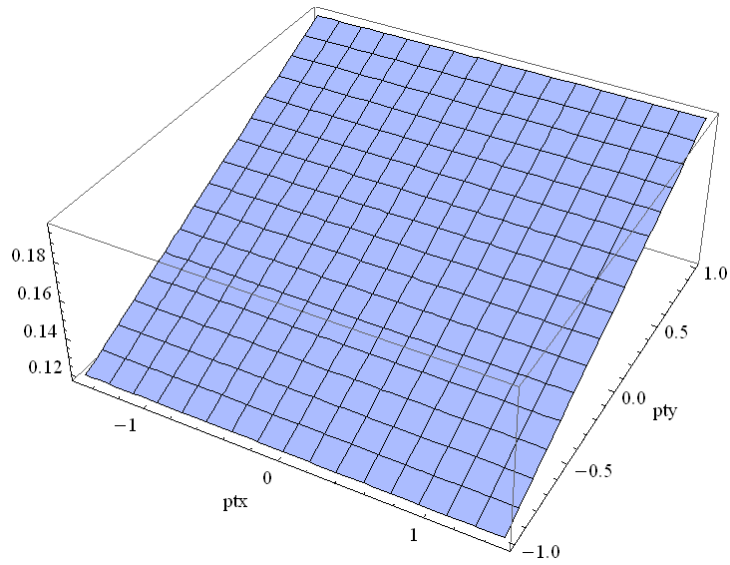
The matrix representation of  $q_i$ :

$$\begin{matrix}
 & & & \text{ptx} \\
 \text{pty} \cos(\text{yaw}) - \text{ptz} \sin(\text{yaw}) & & & \\
 \text{ptz} \cos(\text{yaw}) + \text{pty} \sin(\text{yaw}) & & & \\
 & & & 1
 \end{matrix}$$

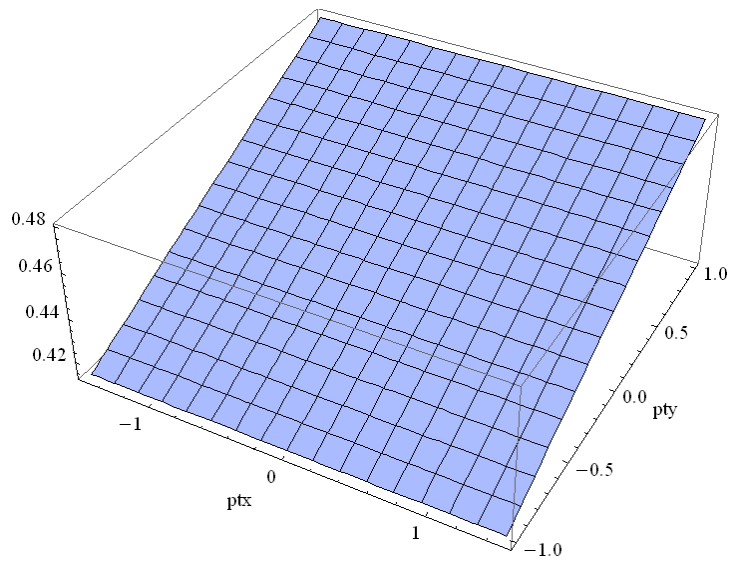


$$f = -\text{ArcTan}[(pty * \text{Cos}[\text{yaw}] - ptz * \text{Sin}[\text{yaw}] - pty) / (c + ptx - ptx)]$$

Yaw=0.5 radians, ptz=1,  $f_{max} - f_{min} \sim 0.08$  radians

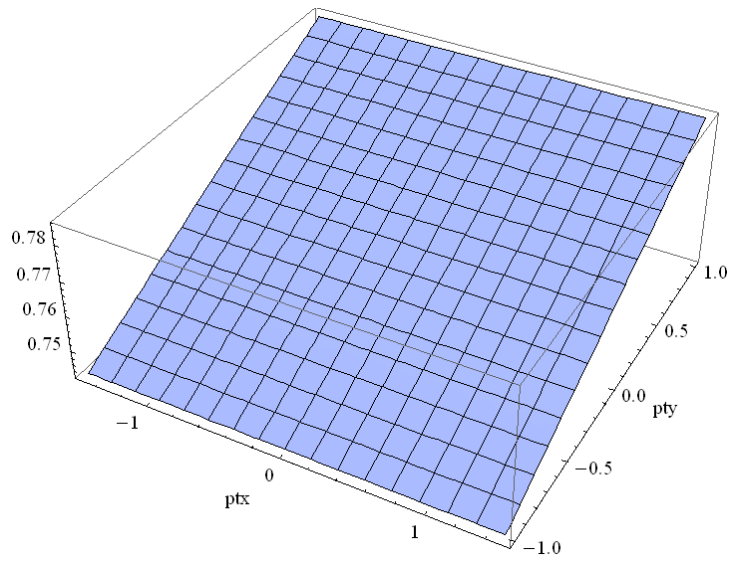


Yaw=0.5 radians, ptz=3,  $f_{max} - f_{min} \sim 0.07$  radians

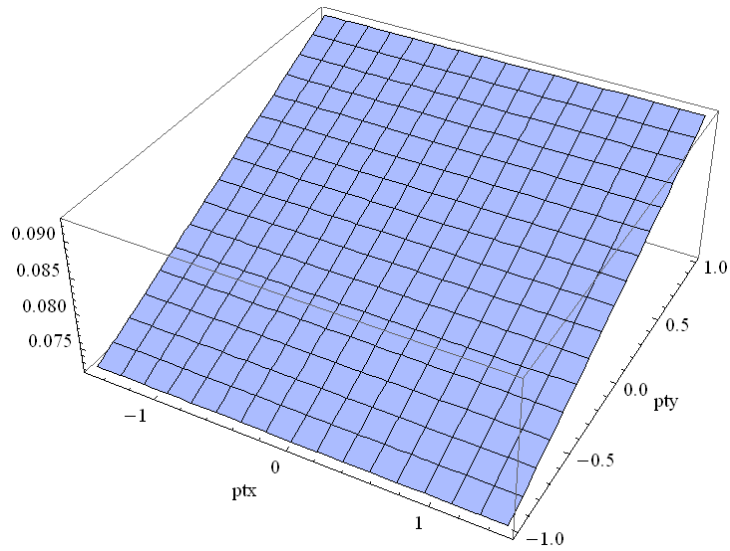


Yaw=0.5 radians, ptz=6,  $f_{max} - f_{min} \sim 0.05$  radians

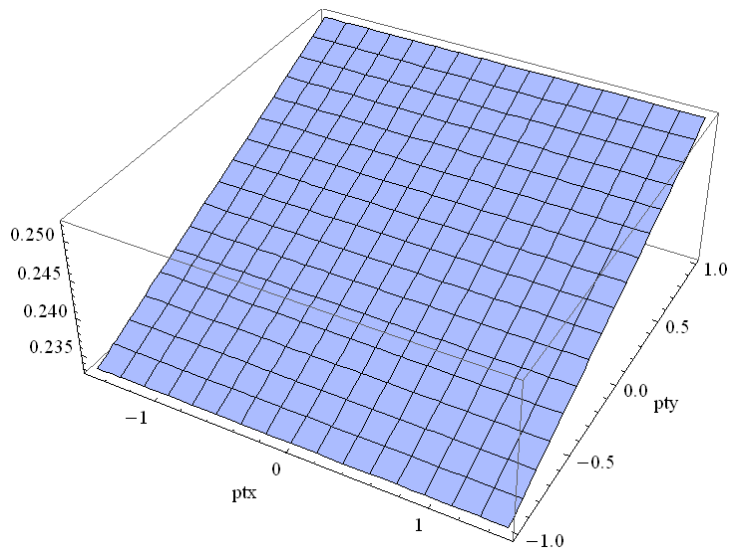




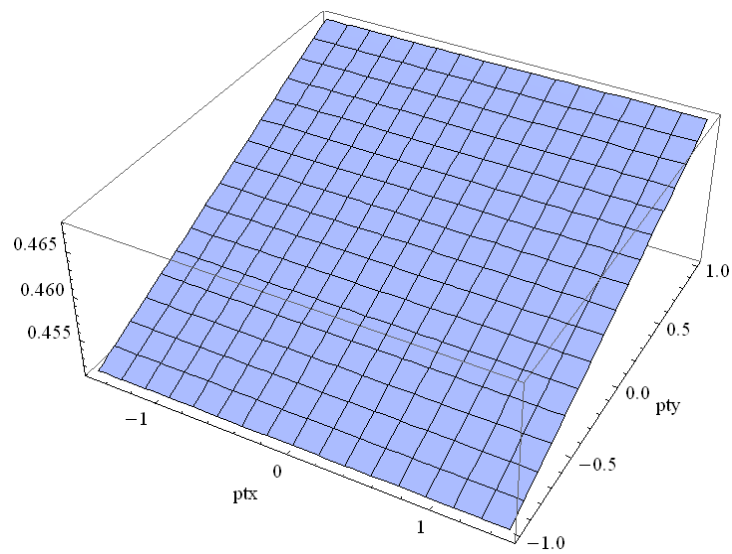
Yaw=0.25 radians, ptz=1 ,  $f_{max} - f_{min} < 0.03 \text{radians}$



Yaw=0.25 radians, ptz=3 ,  $f_{max} - f_{min} < 0.02 \text{radians}$



Yaw=0.25 radians,  $ptz=6$ ,  $f_{max} - f_{min} < 0.015$  radians



#### Case 4,5,6. Effect of translation

Translation along z-axis (depth) will not have effect in  $f$ ; translation along x or y axis will bring about same effect on all  $(p_i, q_i)$  pairs.

#### Tests:

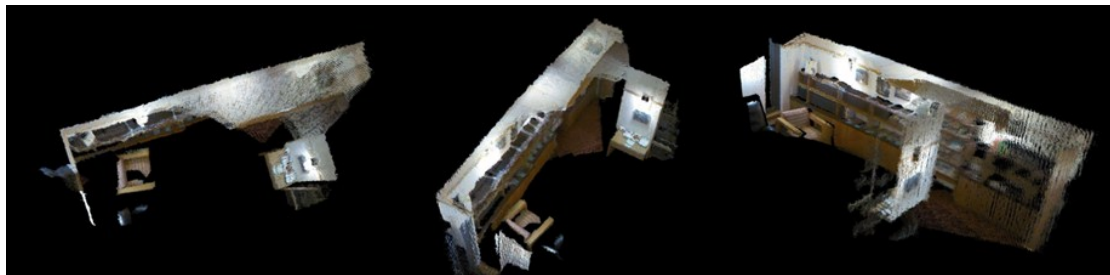


Figure 2.4.2.3.1 Multiple RGBD Images Registration, Result visualized from different angles

## 2.5 Hybrid 2D-RGBD SLAM

### Introduction

Both 2D and RGBD matching methods have their unique advantages and disadvantages. The proposed Hybrid 2D-RGBD SLAM links the two measures and has the following key principles:

- It combines both measures and use each when it works best
- It gives best estimate of the surroundings by learning from multiple observation for long term and short term planning
- It balances and maintains the consistency of observations from the two sensors

### Discussion

According to tests, the 2D SLAM method shown in 2.3 demonstrates its advantages in terms of speed and in general can give more reliable matching and poses estimation. Its major foundation, the probabilistic Bayesian estimation, after an intense research during the last decade [5], turns out to be most successful and can manage well noisy sensory data or uncertainties in robot poses. One the other hand, upon tilt or roll or when robot entering into an unfamiliar environment, the 2D alignment method may fail due to insufficient feature matches.

In comparison, the RGB-D SLAM introduced in 2.4 requires more processing time. Unlike the probabilistic multiple frames fusion discussed in 2.3, the current RGB-D SLAM can only achieve frame-to-frame alignment barely in real time without hardware accelerations such as FPGA or GPU[19] [20] ; thus, due to the limited amount of sensory samples, it may potentially produce less accurate mapping or poses estimation. However, the RGB-D method has unique advantages in perceiving and matching the images features. In terms of associating features among consecutive frames, this measurement is in general more robust and much more insensitive to tilt, roll or changing surroundings observed by the robot.

In terms of modeling the environment for path planning or action planning, most popular approaches include occupancy grid map, points cloud, landmarks or OctoMap--a 3D probabilistic grid map [21][22] or parameterized functions. Although in outdoor dynamic applications estimating the terrain or world is critical, due to the incomplete 3D data from the RGB-D sensor, accurate reconstruction of 3D models can be extremely difficult and expensive [23].

In terms of the proposed Hybrid system, we have the following considerations:

1. Both alignment subroutines, when called together, should agree and converge to a certain point.
2. Even though one alignment method temporally fails, handled by other routines, the system will continue to function and eventually recover to its most reliable main routine.
3. Each routine should be called when it works best.

### 2.5.1 Proposed Algorithm

#### Definitions:

*KeyFrameRGBD: a list of RGBD key frames*

*KeyFrame2D: a list of 2D laser key frames*

*KeyFramePoses: a list of robot pose estimations that corresponds to the key frames*

*DeltaPoseEst: the estimated location and orientation change between two observations*

*NewPoseEst: the estimated pose with respect to the global map based on the matching between two observations, given the pose that corresponds to the previous observation*

*CurPoseEst: estimated current pose, 2D with x,y and phi*

*CurPose3DEst: estimated current pose, 3D with x, y, z, yaw, pitch roll*

*GKMap: global map fused with all 2D key frames (the top map from 2.3.3)*

*GMap; global map fused with all 2D observations (the bottom map from 2.3.3)*

#### Algorithm 2.5.1.1 Hybrid matching

Input: current RGBD frame  $K_i$  and laser scan  $L_i$

*Step1:*

[2D Matching ratio, NewPoseEst]  $\leftarrow$  Perform 2D matching of ( $L_i$ , GKMap, CurPoseEst)  
(as in 2.3.1)

If matching ratio < matching ratio threshold

Go to step 2

Else

Update CurPoseEst= NewPoseEst, CurPose3DEst= NewPoseEst

If necessary\*, update KeyFrameRGBD, KeyFrame2D, KeyFramePoses by adding  $K_i$ ,  $L_i$  and CurPoseEst

*Step2:*

Find the nearest key frame  $NK_j$  from KeyFrame2D, and the estimated pose KeyPose $_j$  that corresponds to this observation

[2D Matching ratio, NewPoseEst]  $\leftarrow$  Perform 2D matching of ( $L_i$ ,  $NK_i$ , KeyPose $_j$ )

If matching ratio < matching ratio threshold

Go to step 3

Else

Update CurPoseEst= NewPoseEst, CurPose3DEst= NewPoseEst

If necessary\*, update KeyFrameRGBD, KeyFrame2D, KeyFramePoses by adding  $K_i$ ,

$L_i$  and CurPoseEst

*Step3:*

Find the nearest key frame RGBDNK<sub>j</sub> from KeyFrameRGBD, and the estimated pose KeyFramePoses<sub>j</sub> that corresponds to this observation

[Matching confidence after Bad Correspondence Rejection\*, DeltaPoseEst] ← Perform RGBD matching of (K<sub>i</sub>, RGBDNK<sub>j</sub>) (as 2.4.1)

If matching confidence < matching confidence threshold

    Stop or resample new frames of data until next successful matching or start a new map building

Else

    TempPoseEst = DeltaPoseEst + KeyPose<sub>i</sub>

[2D Matching ratio, NewPoseEst] ← Perform 2D matching of (L<sub>i</sub>, GKMap, TempPoseEst) (as in 2.3.1)

If matching ratio > (matching ratio threshold – alpha) or NewPoseEst nearly agrees with previous pose

    Done

    If necessary\*, update KeyFrameRGBD, KeyFrame2D, KeyFramePoses by adding K<sub>i</sub>, L<sub>i</sub> and CurPoseEst

Else

    Stop or resample new frames of data until next successful matching or start a new map building

\*The insertion of new key frames is determined if the translation or rotation between the pose associated with the previous key frame and the currently estimated pose is larger than user defined thresholds.

\*In terms of the proposed RGBD alignment, the Matching Confidence is based on the last step-ICP matching ratio as well as number of correspondences after the Bad Correspondence Rejection.

## **Discussion**

First of all, the proposed algorithm is fusing all observations into a 2D global map, based on the estimated pose according to 2D or RGBD matching routine. The 2D global map, need not to exactly describe the environment of the dynamic, unstructured environment. It only gives the robot a rough estimation of its position and relationship to its target. The path planning should be a feedback system between the global map and current observation, for long term and short term path planning. The robot should constantly keep track of its current position with respect to its target, perform pose correction and eventually reach to its target.

As discussed in 2.3.3, the mapping is learning from multiple observations in real time, instead of just registering key frames. Changes in the surroundings can be reflected in the global map.

The 2D alignment and pose estimation, in general cases is faster, consistent and more reliable. Therefore, the 2D matching serves as the main routine.

On the other hand, when the 2D matching ratio is low, two possible situations can occur: 1. sudden changes in the environment; 2. sudden changes in the laser scan pattern due to tilt or roll.

#### *The usage of buffering key frames and System Convergence*

In terms of case 1, this may happen when, for example, the robot is passing a door. Since the global map (the bottom map as defined in 2.3.3) is learned from the past experience, earlier observations weight more than the new observation. When a new pattern that is significantly different from the learnt model is perceived, the 2D global matching may fail. Therefore, we introduce key frame buffers--- the robot keeps on buffering new observations as key frames whenever a small pose change has been made. Note, the updating criterion of adding key frames is different than the top map (as defined in 2.3.3). In general, the update of key frames is more frequent. If the matching ratio between the current observation and global map is low, the robot can perform matching with the buffered frames. Since key frames of observations are continuous, the matching between the current frame and previous key frame is less sensitive to changes in environment. However, this estimation can only be used as a temporal solution, since frequent frame-to-frame alignment can accumulate errors and may cause the system to eventually diverge. If the matching with buffers is successful, new observation must be fused into the global map. Over the next several iterations, new patterns of the surrounding will be learned and weight more in the global map, recovering the 2D global matching routine.

#### *Linking RGBD and 2D matching routine—about Efficiency and Convergence*

Upon significant tilt or roll, even the buffering technique may fail, due to the global changes in sensory data instead of partial changes. In step 3, we first perform RGBD matching against previous RGBD key frame. However, the estimated pose cannot be directly used, since errors may occur in this non-linear matching step due to \ the image perspective distortion, motion blur or sensory data noises. Based on the estimated pose change, we rerun the 2D matching routine to verify and perform adjustment. If matching is successful, new frames will be added to the global map as well as key frame buffers. In general, for the next matching, the buffering technique will be called instead and the system will learn new observations for a few iterations and finally converge to its main routine.

**Example: A calling sequence of the three sub-routines upon a significant tilt at frame 11**

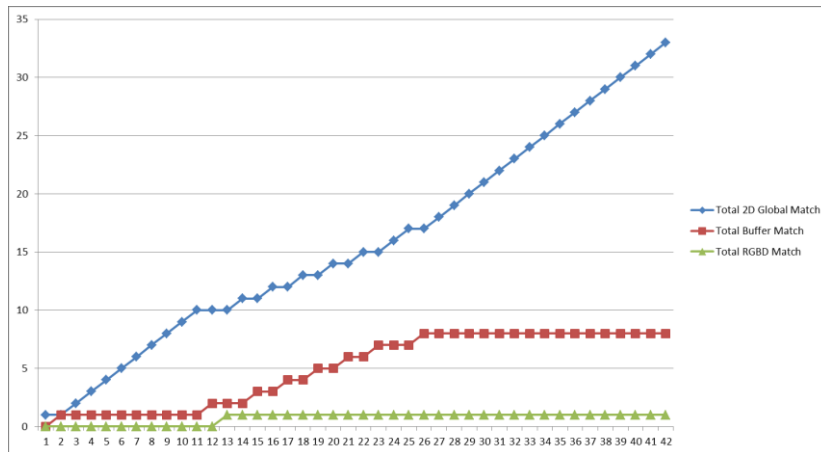


Figure 2.5.2.1 Upon significant tilt at frame 11, the global and buffer matching routines are first being called, but it is handled by the RGBD matching; after this, the system encounter temporal failures of 2D global matching, and the buffer matching has been called until the system eventually converge at the sixth iteration.

**Tests**



Figure 2.5.2.2 The mapping process, all information fused into a 2D global map

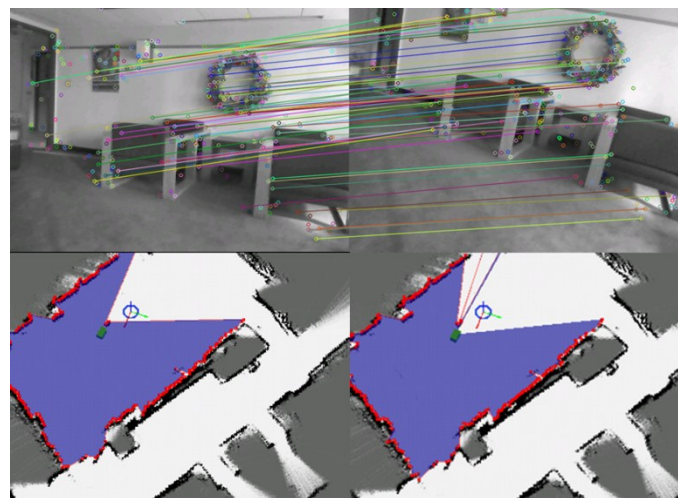


Figure 2.5.2.3 Upon significant tilt and roll, RGBD based matching routine is called, robot pose(green) estimation keeps consistent as before

# Chapter 3. Image Segmentation and Object Classification

In order to understand litters' distributions and forming patterns among reserves, the robot must extract interesting objects from the unknown scenes, and distinguish garbage among other natural objects.

## 3.1 Segmentation with Texture Learning and energy minimization

Assuming most garbage is sitting on the ground, extracting and subtracting the ground component will be helpful for clustering and extracting other interesting objects. The proposed segmentation methods involve computation on color and geometric features, generalization and analysis of texture, energy minimization for optimal solution. It learns about the texture of new environment for the purpose of refinement and more intelligent segmentation among subsequent RGBD images.

Sections 3.1.1 to section 3.1.3 are literature reviews on the most relevant studies to the development of the proposed method in the following aspects: clustering, energy minimization and texture. The reviews are designed to be short but those literatures form the fundamentals of the proposed algorithm.

### 3.1.1 Literature reviews on the most relevant: a. Clustering with Connected Components Labeling

Connected-Component labeling is used in image processing to detect connected regions. In terms of binary image, it detects connected components with same pixel values, and place unique identical labels on those connected pixels. In terms of RGB or RGBD images, by redefining the 'connectivity', this algorithm can be used for clustering pixels with similar features. Trevor et al. [24] proposed an application of connected component in the segmentation of RGBD images.

In general, the connected component labeling has two passes. On the first pass, we iterate through each pixel by column, then by row. For each pixel, we examine its neighbors (the connectivity check is usually with the West and North neighbors). If the current pixel has neighbors, assign the current pixel with the smallest label from its neighbors and store the equivalence between the two labels using data structures such as union-find [25]. If no



neighbor is found, put a unique label on the current element. On the second pass, scan the image again and update the elements with the lowest equivalent label.

As discussed in [24], unlike traditional point clouds, the organized data structure of RGBD images allows a very efficient searching of nearest neighbors. The connectivity between neighboring points is determined by the point normal distance as well as the Euclidean distance.

According to [24], the connectivity  $C(p_1, p_2) = (\text{dist-normal} < \text{thresh-normal})$

We can also combine color features:

$$C(p_1, p_2) = (\text{dist-normal} < \text{thresh-normal}) \parallel (\text{dist-normal} < \text{thresh-normal} + \alpha \parallel \text{color-dist} < \text{thresh-color})$$

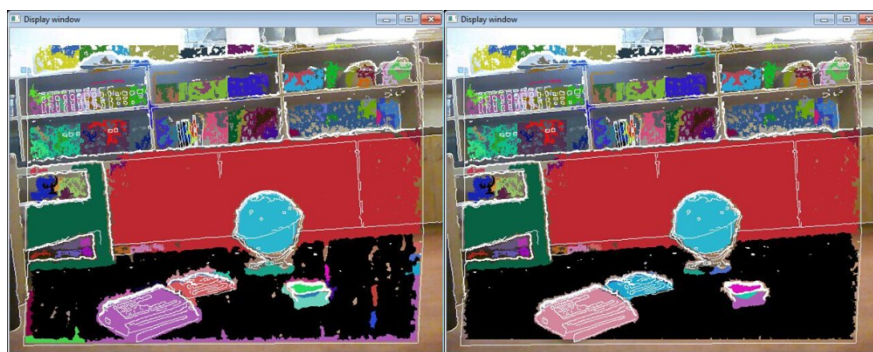


Figure 3.1.1.1 Segmentation using connected component labeling according to similarities of point normal (left); Segmentation by considering the point normal distance, Euclidean distance and color distance.

### 3.1.2 Literature reviews on the most relevant: b.Energy Optimization

The study in energy optimization was initially inspired by the study in stereo image segmentation by [26], which transforms the image segmentation into finding the equilibrium states of the energy of the ferromagnetic Potts model in the super-paramagnetic phase[27][28][29]. According to [26], each pixel was considered as spin (granular ferromagnet) while the Potts model represents a system of interacting spins. Each pixel has  $q$  possible states; the total system energy is the sum of the interaction strength (characterized by color difference) between the neighbors that are only under the same state (described as aligned).

The algorithm [26] aims to iteratively change the state assignment of each spin until it reaches to a configuration with lowest total energy. According to [27], when multiple regions of aligned spins coexist with minimum possible total energy, the system reaches to its equilibrium state, and the regions of aligned spins correspond to the desired image segmentation.

The method proposed by [26] was achieved with GPU based parallel computing. On the other hand, such algorithm may require significant amount of iterations to coverage and may not be feasible for nowadays CPU.

However, the concepts of equilibrium, system energy minimization and its relation to the interaction strength between pixels play a crucial role for the later development.

### 3.1.3 Literature reviews on the most relevant: c.Texture

What is texture? For this research, texture is considered as one generalized global feature that allows us to distinguish one object from the natural scene from human's perspective. Texture is commonly characterized by repeating pattern with randomness. As proposed in [30], the representation of texture involves the modeling of structure and color regularities. Considering common objects from natural scenes, structure regularities may not be a common feature; therefore, this study mainly focuses on the generalization of the color patterns.

Assuming the color texture of natural objects as sets of pixels following natural distributions, the Gaussian Mixture Model is one best fit model. GMM has the following major advantages: 1. It does not require strong independence between features [30]; 2. No prior knowledge or assumption is required for initializing the GMMs [31]; 3. The GMM has many degrees of freedom to update its modeling parameters such as via E-M [32] or K-Means; 4. The training is computationally inexpensive.

The probability density of the GMM is defined as [32] [33]:

$$p(x) = \sum_{j=1}^k \alpha_j p(x | j) \quad (3.1.3.1) [32] [33]$$

where  $\alpha_j$  is the weight of the  $j$ th component

and

$$p(x | j) = \frac{1}{(2\pi)^{d/2} |\Sigma_j|^{1/2}} e^{-1/2(x-\mu_j)^T \Sigma_j^{-1} (x-\mu_j)} \quad (3.1.3.2) [32] [33]$$

Since  $p(x)$  is a probability density function, it not only characterized the mixture distribution of the Gaussian components but can also be used to estimate the fitness of a feature vector input to this mode for the purpose of energy estimation or classification.

### 3.1.4 Proposed RGBD Segmentation with One-Shot Texture Learning and Refinement

The segmentation method proposed here, is assuming interesting objects—the garbage are usually sitting on the ground surface and only need to be extracted from the ground component. The proposed method, however, does not require even ground surface and can be used for more general cases in natural environment. It contains three passes, firstly it estimates the point normal and by clustering techniques gives an approximation of the ground component with high certainty; given the data representing the estimated ground surface, it learns about the texture of the ground component and refine by energy minimization; finally after excluding the refined ground component, it extracts the object clusters nearby the ground and again perform texture learning and refinement for each of them. Based on experiments, only one iteration of refinement is needed; more iterations may potentially cause over-fit problem.

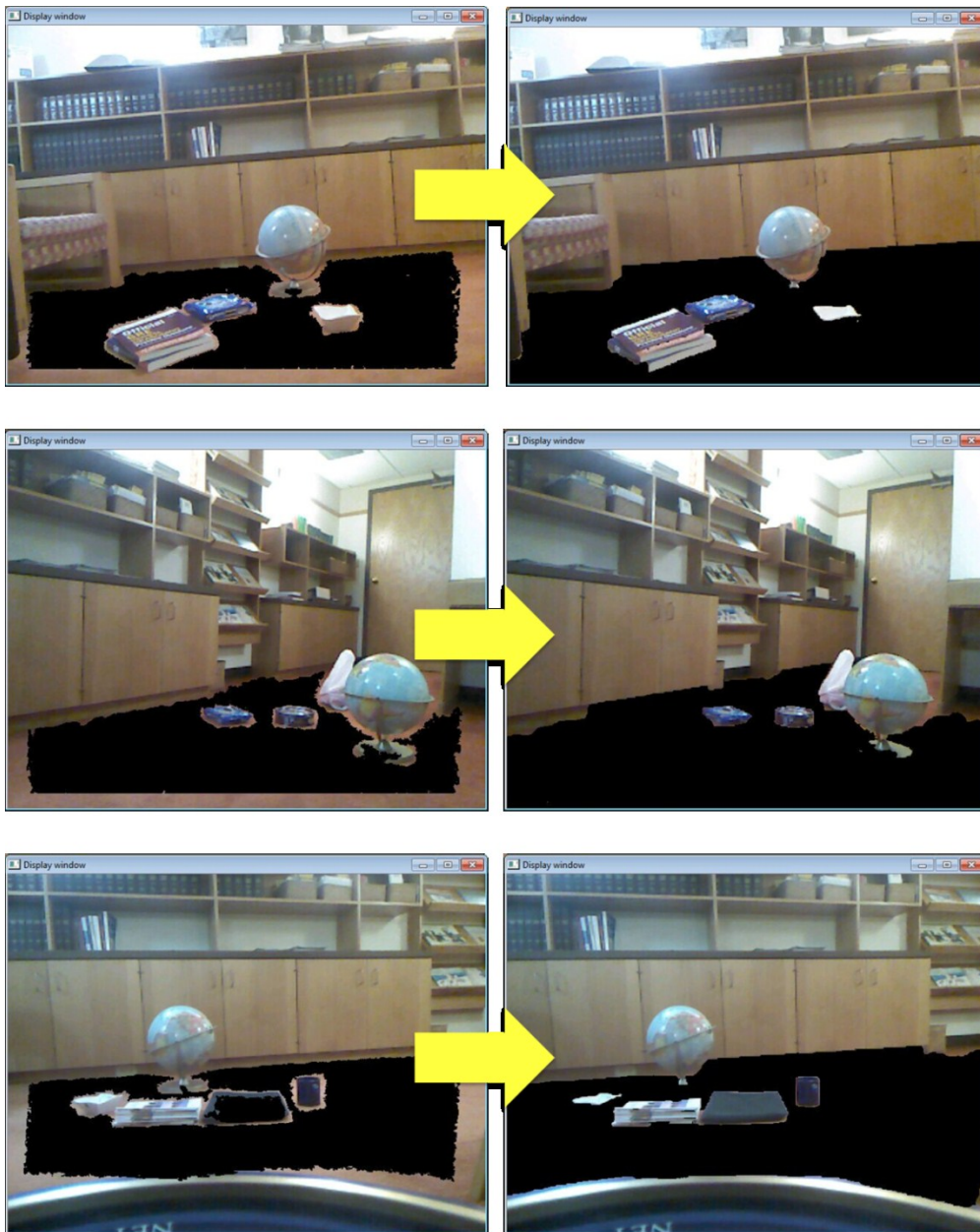
#### Algorithm 3.1.4.1 Ground Component Segmentation with Texture Learning

- Step1:* Estimate surface normals using Integral Images [34] based on covariance matrices.
- Step2:* Use connected component labeling, estimate and extract the ground component  $P = \{p_i\}$ .
- Step3:* Estimate the bounding region of the ground component, get  $ymin, ymax, xmin, xmax$ . Find all non-ground points  $Q = \{q_i\}$  within the bounding box.
- Step3:* Apply K-Means to find K major color components  $C = \{c_1, c_2 \dots c_k\}$  of Q,  $D = \{d_1, d_2 \dots d_k\}$  of P.
- Step4:* Create two GMMs, one for the ground component (GGMM), another for non-ground (NGGMM). Initialize the GMMs with Gaussian Components C and D.
- Step6:* Use P to train the GGMM. Use Q to train the NGGMM. At the same time, compute covariance and inverse covariance matrices and means.
- Step7:* Iterate through all  $p_i$  and  $q_i$  again, for each of them, find the most likely mixture component that maximizes Eq. 3.1.3.2. Retrain GGMM and NGGMM according to the new assignments.
- Step8:* Construct a graph with two terminal nodes representing the ground and non-ground components; add all points from Q as non-terminal nodes. Use Max-flow Min-cut Algorithm to find the minimal energy of graph which represents the whole system, and find the optimal classifications of  $q_i$  according to [35] [36].

Note that step 2 is a rigorous selection of the ground component points. Those points must be determined as part of the ground component with high certainty and won't be considered for the refinement; the points that cannot be determined from the first pass can be classified by the refinement process. The refinement process cannot consider all points as candidates;

otherwise, the result may be unexpected or require unnecessary extra iterations. In step 8, the energy function refers to the Gibbs Energy [35][36], which is characterized by the total interaction strength between the terminal nodes and non-terminal nodes, and within the non-terminal nodes.

## Results



**Step 2. The 'ground' from the first pass**

**Step 3. After one iteration of refinement**

Figure 3.1.4.1.2 Ground component refinement, before & after

### *Cluster Refinement*

In terms of the refinement process of object clusters, due to the limited amount of pixels from the cluster, we run through a more rigorous selection on the refinement candidates.

1. Determine those points that belong to the object with strong confidence.
2. Also determine those points that do not belong to the object (within a bounding box that is larger and enclose the entire object). (Such points can be found by examining the geometric relationship between the object cluster and its surroundings)
3. Train and get GMMs of the object and its background.
4. Extract the points that nearby the boundary of the cluster.
5. Get those points that are not seen by the 3D sensor (i.e. the xyz are undefined)
6. Points from 4) and 5) will be classified based on the two GMMs.

Not all points nearby the object can be classified based on the two GMMs; for example, in some cases the texture of the background and the texture of the object can be very similar; thus, must determine the points that don't belong to the object based on the geometric features and exclude them.

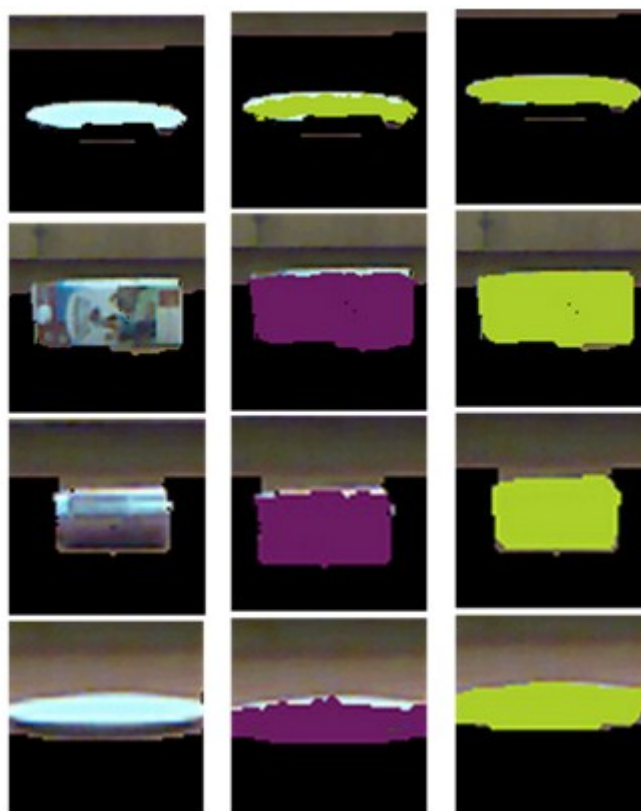


Figure 3.1.4.1.2 After subtracting the ground component, Plate, Milk box, Can were extracted and refined by texture analysis





### 3.2.1.1 Classification using Naïve Bayes Classifier

The Naïve Bayes Classifier has strong assumptions on the independence between features. In terms of texture classification, in most real world cases, this assumption may not be correct. Examples of using Naïve Bayes models for image retrieval and object recognition can be found at [37][38].

An advantage of the Bayes Classifier is that it does not require extensive amount of training. And rather than indicating how likely a given object belongs to class  $C_k$ , it computes the conditional probability according to the feature values among different classes and decide which class is most probable. Considering this, the purpose of using Naïve Bayes Classifier in this study is not to measure the distance between a given object and generalized models, but to compare an object to all class models and determine whether it is most likely to be garbage.

Since the Naïve Bayes Network assumes independence between features, in terms of the feature vector of color textures, we made a strong assumption that the color schemes of artificial products (garbage) are generally chosen around the following major colors: red, orange, yellow, green, cyan, blue, magenta, black and white. Therefore the feature vector contains 9 bins, corresponding to the 9 major colors described above.

Before the color mapping, we transform the pixel value from RGB to HSV. For major colors except black and white, they are quantized based on Figure 3.2.1.1. Pixels with low V (<20%) are considered as black; pixels with low S (<10%) and high V (>90%) are considered as white. Below is the division of the H values:

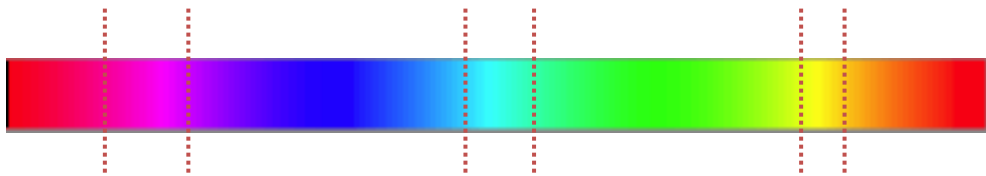


Figure 3.2.1.1 Division of the Hue Values, Red: 335-10; Orange: 11-48; Yellow: 49-67; Green: 68-150; Cyan: 151-206; Blue: 207-259; Magenta: 260-334.

Given an extracted object cluster, we iterative through all pixels, perform quantization according to the mapping shown above, and increment the counts of the corresponding bin. Once the iteration is done, we will end up with a feature vector  $V = \{v_1, v_2 \dots v_9\}$ .

The training is essentially the same, for any sample that belongs to class  $C_k$ , we can generate a new feature vector  $V' = \{v_1', v_2' \dots v_9'\}$ , and add each feature value to the corresponding  $v_k$  at the feature vector  $V_k$  of  $C_k$ .

The 'distance' between a given feature vector  $V$  and  $V_k$  is denoted as

$$D_k = \sum_1^9 (v_i * \text{Log}[v_{ki}/S]), \text{ where } S = \sum_1^9 (v_{ki}) \quad (3.2.1.1)$$

Our goal is to find the class that can give the largest  $D_k$ .

Below are the tests, surprisingly this simplified model works well in many cases:

### 3.2.1.2 Texture Classification Using Gaussian Mixture Model

Another way to think about the texture distance is measure the ‘energy’ of a given object within a Gaussian Mixture Model representing class  $C_k$ . Less energy indicates higher stability.

For all pixels  $P = \{p_1, p_2 \dots p_n\}$  from a given object  $C$ , and a Gaussian Mixture Model with  $k$  components, the energy in this study is denoted as negative log sums of equation 3.1.3.1:

$$\sum_{i=1}^n -\text{Log}[\sum_{j=1}^k p(p_i|j)] \quad (3.2.1.2)$$

where  $p(p_i|j)$  is given by equation 3.1.3.2.

Upon testing, this method works well in many cases as well. Please note that this measurement does not compute the real texture distance, it is rather used for classification as described in the previous section.

### 3.2.2.3 Another Example: Using Earth Mover’s Distance for Texture Distance Measurement

The Earth Mover’s Distance [39], computes the efforts of transforming one feature histogram to another. In this case, features are not required to be independent. Unlike the two methods proposed in the previous sections, it is a measure of distance. Therefore, it can be directly used to form a clustering pattern of samples.

Below is an example showing the texture clustering pattern.

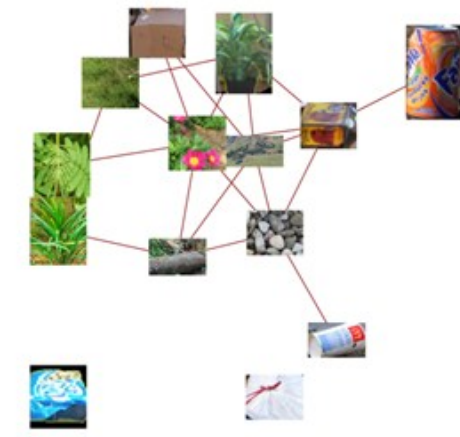


Figure 3.2.2.3.1 Texture Clustering Pattern Example Using EMD, Natural vs. Garbage



# Tests

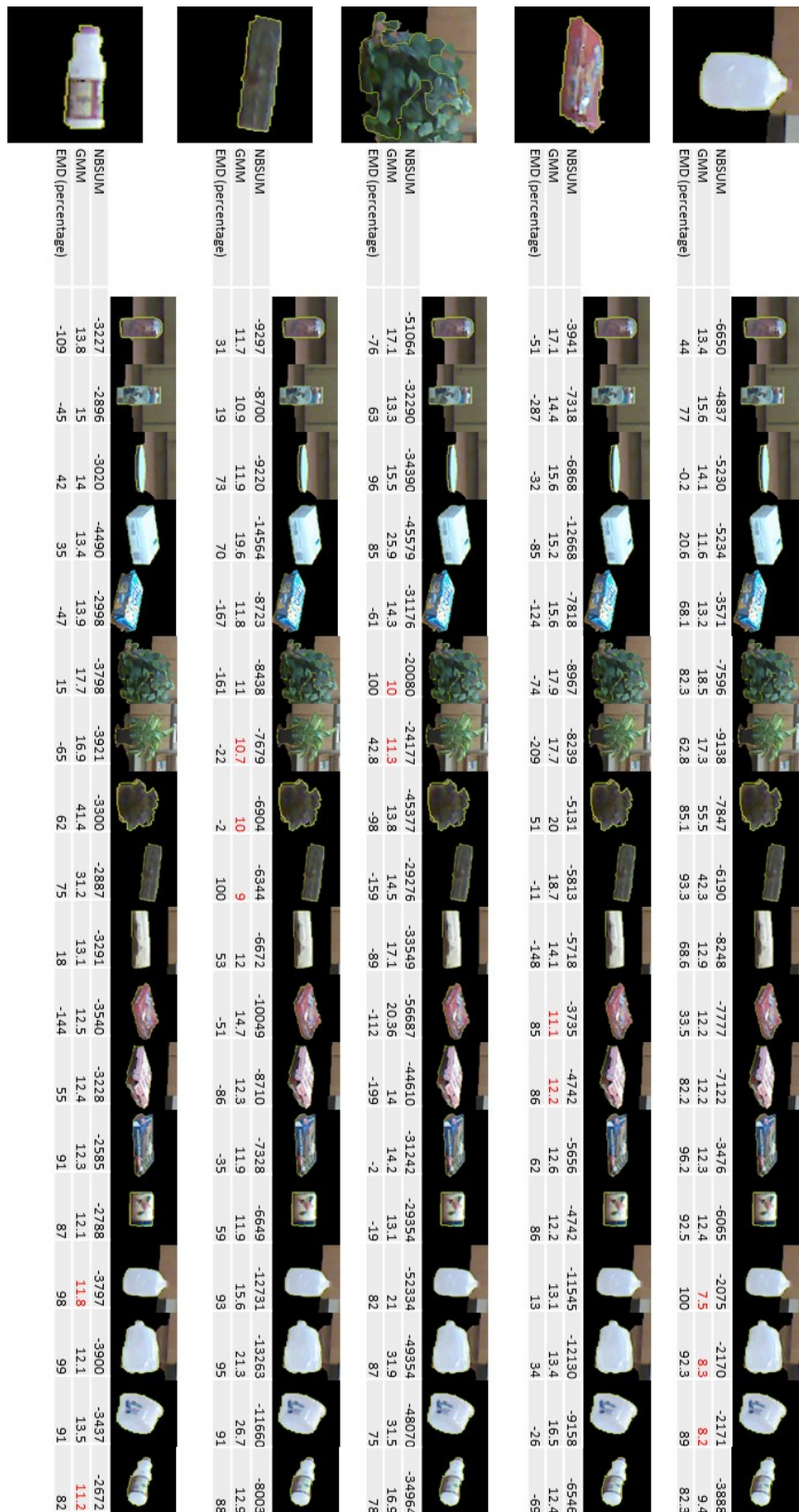


Figure 3.2.1 Texture Classification Results, Naïve Bayes vs. GMM Energy vs. EMD

### 3.2.2.4 Studies on Invariant Shape Descriptors

Besides texture analysis, this study also involves research in generalized shape descriptors. In many cases, the shapes of natural objects that may be seen by the robot such as plants, flowers or trees may be flexible, while garbage such as cans, bottles, boxes or plates in general can have more regularized shapes and may be approximated with cylinder, cube or other general models. Considering changes in view angles, distortion on shapes or imperfect segmentation, we seek to find non-linear solutions.

Although 3D data can be matched against standardized shape models using RANSAC [41], this approach is not directly used considering the incomplete data and noises provided by the Kinect RGBD sensor while RANSAC requires sufficiently enough data.















































































































In terms of 2D, two best known methods are the shape context [42] and Hu's Moments [40]. The 2D matching method must deal with the shape distortion due to changes in view angles. In terms of this, the shape context may be a great choice. The Hu's moments are translation, rotation and scale invariant. While training, by taking sample of objects at different view angles, the distortion problem may be reduced as well. On the other hand, in many cases, we are not performing accurate recognition of the same object, but aims to focus on the matching between a given shape and generalized models.

For the current progress of the research, considering computation efficiency and the reasons described above, we choose Hu's seven Moments. However, as proposed by [43][44], the third Hu's Moment is not independent and not used.

The distance measurement between moments is as follows:

$$\sum_{l=1}^6 \left| \frac{1}{m_{Ai}} - \frac{1}{m_{Bi}} \right| \quad (3.2.2.4)$$

#### Tests

																						
Shape Moments Dist	0.26	0.46	1.88	0.26	0.39	0.49	0.22	0.17	1.02	0.97	0.46	0.64	1	0.02	0	0.05	0.18	0.59				
																						
Shape Moments Dist	0.38	0.23	1.26	0.43	0.28	0.72	0.65	0.8	0.38	0.35	0.19	0.64	0.39	0.64	0.66	0.7	0.82	0.25				
																						
Shape Moments Dist	0.34	0.5	1.98	0.5	0.49	0	0.31	0.39	0.93	1.01	0.36	0.68	0.99	0.33	0.49	0.44	0.42	0.49				
																						
Shape Moments Dist	0.77	0.59	0.86	0.79	0.66	0.94	0.92	1.16	0	0.06	0.58	0.42	0.06	1.02	1.02	1.05	1.2	0.51				
																						
Shape Moments Dist	0.45	0.36	1.29	0.57	0.45	0.68	0.66	0.9	0.3	0.36	0.27	0.21	0.32	0.71	0.79	0.78	0.97	0.23				

### **3.3 Proposed Training and Classification method (In progress)**

In the previous sections, the tests show the potential and discriminative power of the texture and shape classification methods proposed or utilized in this study. In order to classify garbage among the others, we first must define what garbage is, what non-garbage is.

In the context of natural reserves, we consider garbage as a class of various kinds of objects such as cans, plates, plastic bags that are disposed in the environment. While training is currently in progress, the texture clustering pattern (3.2.1.3) sheds lights on the classification and shows it may not just be a problem of “garbage vs. non-garbage”, but also “natural vs. non-natural”. Also, it indicates that the texture of natural objects may potentially form into several major clusters. Moreover, the shape classification tests from the previous section help to illustrate that since garbage such as cans, plates, boxes and bottles may have regularized shapes while natural objects such as plants, trees, flowers in many cases do not, the shape classifier may be useful to give us more confidence about whether a given object that does not have close enough texture features to natural objects is even closer to the major shape models of garbage.

The proposed training and classification method involves two three steps: first, extract the feature histograms of natural and garbage samples and divide the pattern into K major clusters [45]; second, for a given object, compute its minimum texture and shape distances to the cluster centroids from the two main classes, natural and garbage. Finally, pass the minimum distances to another classifier such as neuron network, SVM or decision trees which can give us an estimate of the most probable classification according to the global measures.

The training and the verification of the proposed method is currently in progress.

## **Future Works: On the way to Self-learning and Knowledge Transmission across Generations**

Considering the robot's potential in learning and handling uncertainties in unknown environment, will it be possible to allow a robot to explore and learn by itself?-- The self-learning discussed here, is a learning process that can be achieved without any user defined feature space--the feature space is rather defined and refined by the robot based on its goal and experience.

The purpose of studying self-learning is in the search of learning and passing knowledge across generations. Unlike many other creatures, a robot may be able to explore and transfer the knowledge across generations of other life beings. What information can a robot derive from the literature of our history? What can a robot learn from its own exploration and observation over the next few generations? Can robot sheds lights on our puzzles or help us to better understand ourselves and the universe?

Yet, despite the intelligence and potential of the robot that will be shown in the future development, there is no match between a robot and a real life. Each individual life is such a precious, unique entity that can reason, sense and express its feelings; it comes from the nature, and will eventually return to the nature; once life is over, there will be no repeat.

Creating robots, not only involves the techniques in making machines, but our soulfulness, and the respect for the life in all beings.

# References

- [1] Elfes, A. (1989). Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6), 46-57.
- [2] Bourgault, F., Makarenko, A. A., Williams, S. B., Grocholsky, B., & Durrant-Whyte, H. F. (2002). Information based adaptive robotic exploration. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on* (Vol. 1, pp. 540-545). IEEE.
- [3] Sáez, J. M., & Escolano, F. (2005, April). Entropy minimization SLAM using stereo vision. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on* (pp. 36-43). IEEE.
- [4] Moravec, H. P. (1988). Sensor fusion in certainty grids for mobile robots. *AI magazine*, 9(2), 61.
- [5] Claraco, J. L. B. (2009). Contributions to Localization, Mapping and Navigation in Mobile Robotics. Universidad de M'alaga, M'alaga.
- [6] Segal, A., Haehnel, D., & Thrun, S. (2009, June). Generalized-ICP. In *Robotics: Science and Systems* (Vol. 2, No. 4).
- [7] Zhang, Z. (1994). Iterative point matching for registration of free-form curves and surfaces. *International journal of computer vision*, 13(2), 119-152.
- [8] Stachniss, C., Grisetti, G., & Burgard, W. (2005, April). Recovering particle diversity in a Rao-Blackwellized particle filter for SLAM after actively closing loops. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on* (pp. 655-660). IEEE.
- [9] Stachniss, C., Hahnel, D., & Burgard, W. (2004, October). Exploration with active loop-closing for FastSLAM. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on* (Vol. 2, pp. 1505-1510). IEEE.
- [10] Savelli, F., & Kuipers, B. (2004, October). Loop-closing and planarity in topological map-building. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on* (Vol. 2, pp. 1511-1517). IEEE.
- [11] Bay, H., Tuytelaars, T., & Van Gool, L. (2006). Surf: Speeded up robust features. In *Computer vision–ECCV 2006* (pp. 404-417). Springer Berlin Heidelberg.
- [12] Bay, H., Tuytelaars, T., & Van Gool, L. (2006). Surf: Speeded up robust features. In *Computer vision–ECCV 2006* (pp. 404-417). Springer Berlin Heidelberg.

- [13] Rusu, R. B., Blodow, N., & Beetz, M. (2009, May). Fast point feature histograms (FPFH) for 3D registration. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on* (pp. 3212-3217). IEEE.
- [14] Grundmann, M., Meier, F., & Essa, I. (2008, December). 3D shape context and distance transform for action recognition. In *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on* (pp. 1-4). IEEE.
- [15] Tombari, F., Salti, S., & Di Stefano, L. (2010). Unique signatures of histograms for local surface description. In *Computer Vision–ECCV 2010* (pp. 356-369). Springer Berlin Heidelberg.
- [16] Steder, B., Rusu, R. B., Konolige, K., & Burgard, W. (2010, October). NARF: 3D range image features for object recognition. In *Workshop on Defining and Solving Realistic Perception Problems in Personal Robotics at the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS) (Vol. 44)*.
- [17] Henry, P., Krainin, M., Herbst, E., Ren, X., & Fox, D. (2010). RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments. In *In the 12th International Symposium on Experimental Robotics (ISER)*.
- [18] Huber, P. J. (2011). *Robust statistics* (pp. 1248-1251). Springer Berlin Heidelberg.
- [19] Se, S., Ng, H. K., Jasiobedzki, P., & Moyung, T. J. (2004, October). Vision based modeling and localization for planetary exploration rovers. In *Proceedings of International Astronautical Congress* (pp. 434-440).
- [20] Sinha, S. N., Frahm, J. M., Pollefeys, M., & Genc, Y. (2006, May). GPU-based video feature tracking and matching. In *EDGE, Workshop on Edge Computing Using New Commodity Architectures (Vol. 278, p. 4321)*.
- [21] Hornung, A., Wurm, K. M., Bennewitz, M., Stachniss, C., & Burgard, W. (2013). OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, 34(3), 189-206.
- [22] Wurm, K. M., Hornung, A., Bennewitz, M., Stachniss, C., & Burgard, W. (2010, May). OctoMap: A probabilistic, flexible, and compact 3D map representation for robotic systems. In *Proc. of the ICRA 2010 workshop on best practice in 3D perception and modeling for mobile manipulation (Vol. 2)*.
- [23] Hadsell, R., Bagnell, J. A., Huber, D. F., & Hebert, M. (2009, June). Accurate rough terrain estimation with space-carving kernels. In *Robotics: Science and Systems (Vol. 2009)*.

- [24] Trevor, A. J., Gedikli, S., Rusu, R. B., & Christensen, H. I. (2013). Efficient organized point cloud segmentation with connected components. *Semantic Perception Mapping and Exploration (SPME)*.
- [25] Linda Shapiro and George C Stockman. *Computer Vision*, chapter 3, pages 69–75. Prentice Hall, 2001.
- [26] Abramov, A., Aksoy, E. E., Dörr, J., Wörgötter, F., Pauwels, K., & Dellen, B. (2010). 3d semantic representation of actions from efficient stereo-image-sequence segmentation on GPUs.
- [27] Blatt, M., Wiseman, S., & Domany, E. (1996). Superparamagnetic clustering of data. *Physical review letters*, 76(18), 3251.
- [28] Opara, R., & Wörgötter, F. (1998). A fast and robust cluster update algorithm for image segmentation in spin-lattice models without annealing—visual latencies revisited. *Neural Computation*, 10(6), 1547-1566.
- [29] von Ferber, C., & Wörgötter, F. (2000). Cluster update algorithm and recognition. *Physical Review E*, 62(2), R1461.
- [30] Permuter, H., Francos, J., & Jermyn, H. (2003, April). Gaussian mixture models of texture and colour for image database retrieval. In *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on (Vol. 3, pp. III-569)*. IEEE.
- [31] Reynolds, D. A., Quatieri, T. F., & Dunn, R. B. (2000). Speaker verification using adapted Gaussian mixture models. *Digital signal processing*, 10(1), 19-41.
- [32] Bilmes, J. A. (1998). A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models. *International Computer Science Institute*, 4(510), 126.
- [33] Gokalp, D. (2005). *Learning Skin Pixels in Color Images Using Gaussian Mixture*. Technical report, Bilkent University, Ankara, Turkey.
- [34] Holzer, S., Rusu, R. B., Dixon, M., Gedikli, S., & Navab, N. (2012, October). Adaptive neighborhood selection for real-time surface normal estimation from organized point cloud data using integral images. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on (pp. 2684-2689)*. IEEE.
- [35] Rother, C., Kolmogorov, V., & Blake, A. (2004). Grabcut: Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics (TOG)*, 23(3), 309-314.

- [36] Boykov, Y., & Kolmogorov, V. (2003, October). Computing geodesics and minimal surfaces via graph cuts. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on* (pp. 26-33). IEEE.
- [37] Boiman, O., Shechtman, E., & Irani, M. (2008, June). In defense of nearest-neighbor based image classification. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on* (pp. 1-8). IEEE.
- [38] Tang, J., Miller, S., Singh, A., & Abbeel, P. (2012, May). A textured object recognition pipeline for color and depth image data. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on* (pp. 3467-3474). IEEE.
- [39] Rubner, Y., Tomasi, C., & Guibas, L. J. (2000). The earth mover's distance as a metric for image retrieval. *International journal of computer vision*, 40(2), 99-121.
- [40] Hu, M. K. (1962). Visual pattern recognition by moment invariants. *Information Theory, IRE Transactions on*, 8(2), 179-187.
- [41] Schnabel, R., Wahl, R., & Klein, R. (2007, June). Efficient RANSAC for Point - Cloud Shape Detection. In *Computer graphics forum* (Vol. 26, No. 2, pp. 214-226). Blackwell Publishing Ltd.
- [42] Belongie, S., Malik, J., & Puzicha, J. (2002). Shape matching and object recognition using shape contexts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(4), 509-522.
- [43] Flusser, J. (2000). On the independence of rotation moment invariants. *Pattern recognition*, 33(9), 1405-1410.
- [44] Flusser, J., & Suk, T. (2006). Rotation moment invariants for recognition of symmetric objects. *IEEE Transactions on Image Processing*, 15(12), 3784-3790.
- [45] Rusu, R. B. (2010). Semantic 3d object maps for everyday manipulation in human living environments. *KI-Künstliche Intelligenz*, 24(4), 345-348.



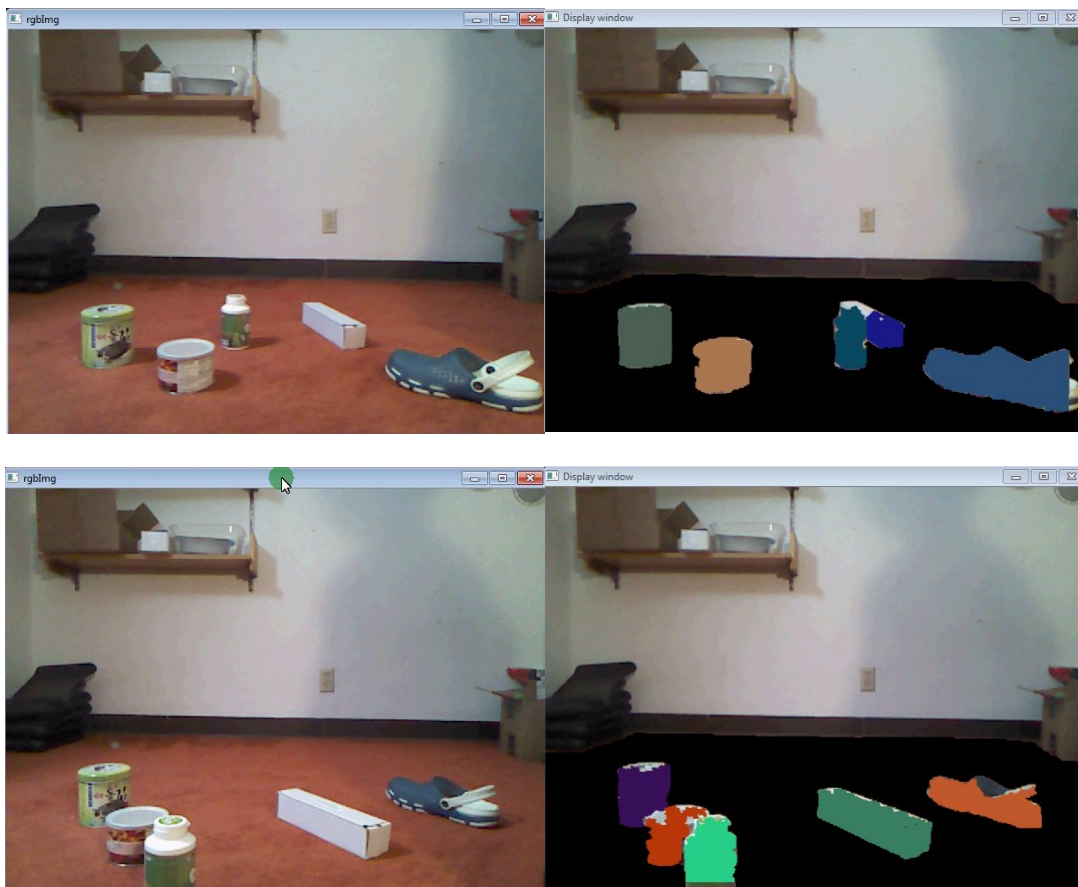
# Appendix

## Video Demos

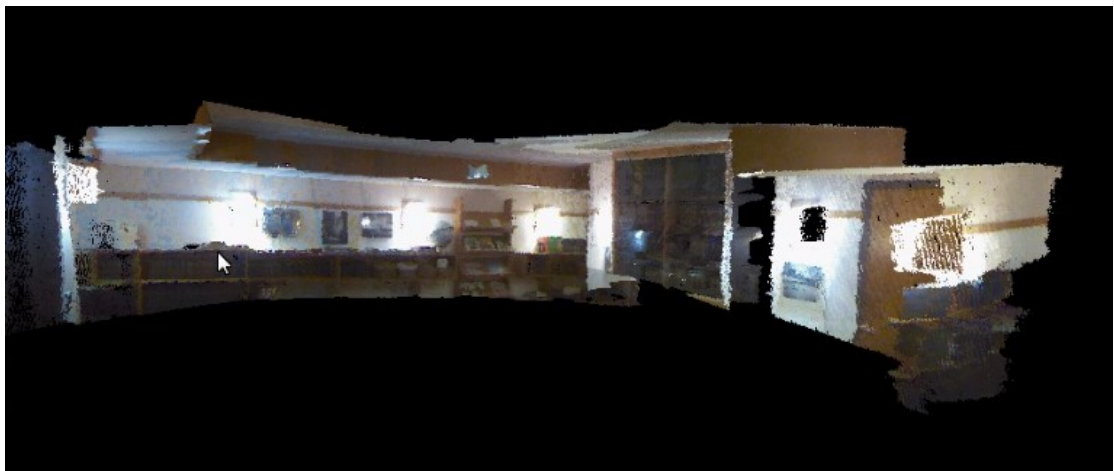
Video Demos can be found at <https://homepages.cae.wisc.edu/~xdeng/simple/xiang.html>. This research will be continued and new videos will be posted on this site in the future.

*Demos include:*

### **Image Segmentaiton with One-shot Learning and Refinement**



## RGBD Registration



## Hybrid SLAM

