

國立交通大學
電機與控制工程研究所

碩士論文

移動式機器人與未知數量聲源之即時同步定位



Simultaneous Localization of Mobile Robot and
Unknown Number of Multiple Sound Sources

研究生：詹鎮宇

指導教授：胡竹生 博士

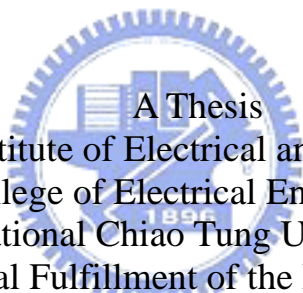
中華民國九十八年六月

移動式機器人與未知數量聲源之即時同步定位

Simultaneous Localization of Mobile Robot and
Unknown Number of Multiple Sound Sources

研究生：詹 鎮 宇 Student: Chen-Yu Chan
指導教授：胡 竹 生 博士 Advisor: Dr. Jwu-Sheng Hu

國立交通大學
電機與控制工程學系
碩 士 論 文



A Thesis
Submitted to Institute of Electrical and Control Engineering
College of Electrical Engineering
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of Master
in

Electrical and Control Engineering

June 2009

Hsinchu, Taiwan, Republic of China

中 華 民 國 九 十 八 年 六 月

移動式機器人與未知數量聲源之即時同步定位

研究生：詹 鎮 宇

指導教授：胡 竹 生 博士

國立交通大學電機與控制工程研究所碩士班

摘 要

本論文提出一套在未知聲源個數的情況下能同步定位移動式機器人平台聲源的方法，並解釋了使用聲音作為定位地標的原因。文中介紹了幾種角度估測(DOA)的方法，並結合其中一些方法的特性來達到多聲源角度估測的目的，這些角度估測將使用在理論基礎為貝氏濾波器(Bayes Filter)的純角度資訊同步定位演算法(Bearings-only SLAM)。因聲源非同步發聲，且聲源沒有角度以外的資訊，故有未知資料群落(Unknown Data Association)的問題需解決，演算法中的粒子濾波器(particle filter)將可處理這部分的問題。本研究為了將演算法即時實現在實驗環境中而做了不影響理論結果的修改，並且展示了實驗成果來證實演算法的效能。

Simultaneous Localization of Mobile Robot and Unknown Number of Multiple Sound Sources

Student : Chen-Yu Chan

Advisor : Dr. Jwu-Sheng Hu

Institute of Electrical and Control Engineering

ABSTRACT

This work proposes a method that is able to simultaneously localize a mobile robot and unknown number of multiple sound sources in the environment. The reason of using sound sources as the landmarks in SLAM algorithm is presented. Several DOA estimation methods are described and a combinational one is used for real time application. After knowing the DOA information, a bearings-only SLAM (simultaneous localization and mapping) algorithm is introduced in detail, which contains the theoretical structure of Bayes filter. The estimated DOAs are known as the bearings information in the algorithm. As source signals are not persistent and there is no identification of the signal content, data association is unknown which is solved using particle filter. Modifications of the algorithm are made for real time application. Experimental results are presented to verify the effectiveness of the proposed approaches.

誌 謝

兩年多的研究生涯隨著這份論文的完成，畫下一個令人不捨的句點。由衷感謝我的指導教授胡竹生博士兩年多來的細心教誨，老師積極進取的態度、深厚的理論基礎、不落窠臼的創新思考，都是我應該要努力追求的目標，也謝謝老師在出國留學上給我許多建議，以及在國際會議時對我的諸多提醒，讓我除了交大的研究生活，更積極接觸到台灣以外的優秀研究成果，得到滿滿的收穫。

感謝實驗室的大家，讓我的研究生活可以多采多姿。謝謝立偉學長帶我進入機器人的領域；實驗室網頁不可或缺的宗敏學長；讓我學習到什麼是品味生活的劉大；總是讓實驗室充滿咖啡香的鏗元；帶領我進入聲音研究群，解決我很多研究上的困惑，跟我一樣希望台灣獨立的興哥；子母機器人被我們拆得差不多的楷祥；在英文與當兵上給我很多建議的法哥；我在實驗室的心靈導師永融，對不起我學不到你一成的功力就要出去闖天下了；事情都可以簡單來說的阿吉；聯誼一哥白馬丸子大師兄；實驗室獨霸一方的女高音瓊文；攻城比龐青雲還快的帥氣小周董 HCY；常常說要灌人後腦的車痴俊宇；獨霸實驗室另一方的男高音 papa，謝謝你，研究少你了可能完全不知道從哪裡下手；車子很好開人很 nice 又很 high 的槓葛格；集合帥氣笑容、專業認真、籃球神經於一身的明唐。再來要謝謝我實驗室的同梯們；神秘感十足其實很愛家但是又很搖滾再加上一點點娘的 Lundy 大大(轉圈圈灑花～)；籃球打很好雖然有時會猶豫不決但其實很認真做研究的肉鬆；同梯的驕傲老師一定很高興你念博班的神拳 judo；將要成為實驗室碩班大弟子祝你可以去阿拉斯加的小蔡；推薦了我很多音樂的衝浪男孩 simon；跟我一樣會溜去打 board game 的沛錡；對耳機小有研究的聖翔；羽球比我強很多的澳門小鋼炮阿 him；介紹很多古典樂給我且很貼心的小朱朱；Rodolfo, é muito bom ter você aqui no laboratório!

謝謝我的家人在我進交大之後就不斷給我鼓勵，嗶爸詹明賢，嗶媽李里美，嗶哥詹適宇，在我承受巨大心理壓力的時候可以支持著我度過難關，最後要謝謝小紅這半年來總是要接受我沒來由的負面情緒；謝謝所有人！交大！暫別了！

CONTENT

摘要	i
ABSTRACT	ii
誌謝	iii
CONTENT	iv
LIST OF TABLES	vi
LIST OF FIGURES.....	vii
Chapter 1. INTRODUCTION.....	1
1.1 Motivation and Objective	1
1.2 Literature Review	2
1.3 Thesis Subject and Contribution.....	3
1.4 Outlines of Thesis	4
Chapter 2. DETECTION OF MULTIPLE SOUND SOURCES.....	5
2.1 Introduction.....	5
2.2 Mathematical Structure of TDE and Eigenspace Method	5
2.2.1 TDE	5
2.2.2 ES-GCC, a combination of TDE and Eigenspace Method	7
2.3 Performance of Different Eigenspace Methods in Real Time Application	11
Chapter 3. BEARINGS-ONLY SLAM ALGORITHM.....	12
3.1 Between DOA estimation and Bearings-Only SLAM.....	12
3.2 Introduction of Probabilistic Robotics.....	12
3.3 Bayes Filter	13
3.3.1 State Estimation using Probabilistic Generative Laws.....	13
3.3.2 Parametric Filter – Kalman Filter and Extended Kalman Filter.....	15
3.3.3 Nonparametric Filter - Particle Filter.....	19
3.4 SLAM	21
3.4.1 Problem Definition.....	21
3.4.2 FastSLAM.....	23
3.4.3 Unknown Data Association.....	25
3.5 Simulation Result of Fast SLAM	25

Chapter 4. Experimental Result and Discussion.....	29
4.1 Introduction of the Robot Platform	29
4.2 Performance of offline calculated algorithm	31
4.3 Performance of online algorithm.....	34
Chapter 5. Conclusion and Future Study	38
REFERENCE	39



LIST OF TABLES

TABLE 1. PARTICLES IN FASTSLAM ARE COMPOSED OF A PATH ESTIMATE AND A SET OF ESTIMATORS OF INDIVIDUAL FEATURE LOCATIONS WITH ASSOCIATED COVARIANCE	24
TABLE 2. THE SPECIFICATION OF THE <i>POINEER 3-DX</i> ROBOT [9].....	29
TABLE 3. THE ERROR ANALYSIS OF OFFLINE CALCULATED FASTSLAM.....	33
TABLE 4. THE ERROR ANALYSIS OF REAL-TIME FASTSLAM.....	37



LIST OF FIGURES

FIGURE 1. MODEL OF A UNIFORM LINEAR ARRAY	9
FIGURE 2. THE PSEUDO-ALGORITHM FOR BAYES FILTERING	15
FIGURE 3. THE KALMAN FILTER ALGORITHM FOR LINEAR GAUSSIAN STATE TRANSITIONS AND MEASUREMENTS	17
FIGURE 4. THE EXTENDED KALMAN FILTER ALGORITHM	19
FIGURE 5. THE PARTICLE FILTER ALGORITHM, A BAYES FILTER BASED ON IMPORTANCE SAMPLING	21
FIGURE 6. THE SIMULATION PROCEDURE OF THE FASTSLAM ALGORITHM	26
FIGURE 7. SIMULATION RESULTS OF THE FASTSLAM ALGORITHM WITHOUT RESAMPLE (ONLY SINGLE FEATURE IS ILLUSTRATED).....	27
FIGURE 8. SIMULATION RESULTS OF THE FASTSLAM ALGORITHM WITH RESAMPLE	28
FIGURE 9. OVERALL PICTURE OF THE MOBILE ROBOT PLATFORM	29
FIGURE 10. BLOCK DIAGRAM OF THE MICROPHONE ARRAY	30
FIGURE 11. (A) THE 8-CHANNEL DIGITAL MICROPHONE (B) THE FPGA AND THE USB MODULE	31
FIGURE 12. THE SPATIAL RELATION OF THE SPEAKER, THE ROBOT, AND THE LASER RANGE FINDER.....	31
FIGURE 13. THE ALGORITHM PROCEDURE FOR OFFLINE CALCULATION	32
FIGURE 14. THE EXPERIMENTAL RESULTS OF OFFLINE CALCULATED FASTSLAM.....	33
FIGURE 15. THE ALGORITHM PROCEDURE FOR ONLINE CALCULATION.....	34
FIGURE 16. THE REAL-TIME EXPERIMENTAL ENVIRONMENT	35
FIGURE 17. THE REAL-TIME EXPERIMENTAL RESULTS	36
FIGURE 18. THE FINAL REAL-TIME EXPERIMENTAL RESULTS.....	37



Chapter 1. INTRODUCTION

1.1 Motivation and Objective

Robots are getting closer to human life in recent years. The interaction between robots and human is a key factor how the robot could be applied to real environment with human around. Among the interaction scenarios between robot and human, localization of the robot is frequently discussed. Lots of the localization methods that have been studied are using camera or laser range finder as the sensor input.

The cameras are able to distinguish different landmarks by using the visual information such as color, object edge, shape of landmarks, etc. The association between different temporal frames of sensor data is feasible. On the other hand, laser range finders are able to provide accurate range information of different landmarks. A more precisely measured data would help the localization procedure converges faster. However, both of these two sensors are suffering from various drawbacks such as occlusion. If a landmark is NLOS (non-line-of-sight), it will be not detectable.

We intend to propose a method focusing on solving the occlusion problem. Acoustic signals are detectable even if a landmark is NLOS. Also, in human environment, there are stationary sound sources such as air-conditioner, operating sound of a computer, and non-stationary sound as human speech, door-knock, etc. According to this characteristic, we consider multiple sound sources as the landmarks in the localization problem. In that sense, a high-dimensional microphone array is mounted on a mobile robot to detect these sound signals, and the robot has its own encoder information to help the self localization of the robot.

1.2 Literature Review

The array signal processing technology [1] was first used in the World War I. The invention was used to detect enemy aircrafts. The technology was applied to array telescope such as the Very Large Array in New Mexico, USA. In this thesis, we use a microphone array to implement the array signal processing algorithm [2].

There are generally two categories of method to solve direction of arrival (DOA) estimation problem. One of the methods is TDE (time delay estimation) proposed by Knapp and Carter [3]. Two microphones are used to record the sound signal, and sound sources from different direction will cause different response to these two microphones. To measure the time delay between microphones, a commonly used method is GCC (Generalized Cross Correlation). Finding the maximum value of the GCC expression indirectly indicates knowing the delay relation. The temporal difference between these two responses could be used to estimate the direction.

The second well-known DOA estimation method is eigenspace method. It measures the distribution of eigenvector between different signals and estimates the signal direction by mutual projection. The MUSIC algorithm proposed by Schmidt [4] belongs to this category. It is able to localize multiple sound sources with prior knowledge of the total number of sources.

After the DOA estimation, we use this information to localize the mobile robot. There are a lot of researches focusing on solving SLAM (simultaneous localization and mapping) problem. Bayes filters [6] are commonly used to both describe the position state of a robot itself and the environmental landmarks position estimation.

1.3 Thesis Subject and Contribution

The subject of this thesis can be divided into two parts. The first part is to implement a multiple sources DOA estimation algorithm. The DOA estimation results will be considered as the bearing measurements of the second part of the thesis. The second part of the method is the Bearings-Only SLAM algorithm that is able to simultaneously localize the features in the experimental environment and to map the robot itself to the environment.

In the first part, we compare the effectiveness of the two algorithms, which are ES-GCC and accumulative MUSIC. ES-GCC is an eigenspace based method that is able to detect unknown number of multiple sound sources direction simultaneously. MUSIC is a more commonly used DOA estimation method that could only estimate known number of multiple sound sources. We monitor the number of time frames needed for both algorithms in real time application, and modified them to satisfy the hardware limitation.

The second part is a SLAM problem architecture, which considers the outputs of the first part as the measurements. SLAM problem could be solved using procedures based on the Bayes filter. The particle filter is used for the non-parametric robot environment. The removal of the resample step in particle filter doesn't disturb the algorithm but decrease the computing complexity and shorten the algorithm procedure so that the method is able to be applied in real time.

The experimental results are shown to justified the valid modifications, yet accomplishes the simultaneous localization and mapping problem in real-time.

1.4 Outlines of Thesis

The remainder of this thesis is organized as follows.

Chapter 2: The two kinds of DOA estimation method are clearly described, including the performance analysis for real time application. The reasons of choosing eigenspace method are explained and there would be modification made to the algorithm to solve the insufficient performance of real time experiment.

Chapter 3: The detail concept of Bayes filter is stated in this chapter. Based on the concept, a bearings-only SLAM algorithm constructed by EKF (extended Kalman filter) and PF (particle filter) is introduced. The mathematical detail is also in this chapter. Finally, PF is able to deal with the unknown data association between different temporal frames.

Chapter 4: The experimental results are presented. A combinational architecture of DOA estimation and bearings-only SLAM is applied to a real mobile robot, and the real time performance analysis is discussed.

Chapter 5: The conclusion of this thesis and the possible improvement in the future is presented in this chapter.

Chapter 2. DETECTION OF MULTIPLE SOUND SOURCES

2.1 Introduction

DOA (direction of arrival) estimation is usually mentioned as the research of estimating the spatial position of signals, or to measure the emitting direction between signals and sensors. In technical points of view, DOA estimation contains two categories, TDE (time delay estimation) and eigenspace method. Although TDE is only able to distinguish the emitting direction of single non-overlapped sound source, it needs only two microphones. The hardware architecture is comparatively simple, and the order of computation is more suitable for real time application. On the other hand, eigenspace method is able to distinguish multiple sound sources with the prior knowledge of the quantity of sources. Even if the sources are temporal overlapped.

2.2 Mathematical Structure of TDE and Eigenspace Method

2.2.1 TDE

Two microphones are used to record the sound signal, and sound sources from different direction will cause different response to these two microphones. To measure the time delay between microphones, a commonly used method is GCC (Generalized Cross Correlation). Suppose there is one single source in the environment, in ideal case the sound signal received by the two microphones could be expressed as

$$x_1(t) = s_1(t) + n_1(t) \quad (2.1)$$

$$x_2(t) = \alpha s_1(t + D) + n_2(t) \quad (2.2)$$

where $s_1(t)$, $n_1(t)$, $n_2(t)$ are WSS (wide sense stationary) and uncorrelated. D is

the real delay between two microphones. α is a magnitude scaling factor. D and α are changing relatively slower than $s_1(t)$, which is the signal itself. The cross correlation between the microphones is

$$R_{x_1, x_2}(\tau) = E[x_1(t)x_2(t-\tau)] \quad (2.3)$$

where E is the expectation value. The related τ that gets maximum $R_{x_1, x_2}(\tau)$ is the delay value. Due to limited observation time, the estimated cross correlation is expressed as

$$\hat{R}_{x_1, x_2}(\tau) = \frac{1}{T-\tau} \int_{\tau}^T x_1(t)x_2(t-\tau)dt \quad (2.4)$$

T represents the observation time interval. The cross correlation is the inverse Fourier Transform of the cross power spectrum. Expressed as

$$R_{x_1, x_2}(\tau) = \int_{-\infty}^{\infty} G_{x_1, x_2}(f)e^{j2\pi f\tau}df \quad (2.5)$$

Consider the real spatial situation, the sound signal received by the microphones are transformed by space, so the cross power spectrum between the real microphones are

$$G_{y_1, y_2}(f) = H_1(f)H_2^*(f)G_{x_1, x_2}(f) \quad (2.6)$$

where $H_1(f)$ and $H_2(f)$ are the spatial impulse response form the signal to the first and the second microphone. So, we define the generalized cross correlation between the microphone pair as

$$R_{y_1, y_2}^{(g)}(\tau) = \int_{-\infty}^{\infty} \psi_g(f)G_{x_1, x_2}(f)e^{j2\pi f\tau}df \quad (2.7)$$

where

$$\psi_g(f) = H_1(f)H_2^*(f) \quad (2.8)$$

Practically, we should substitute $G_{x_1, x_2}(f)$ by $\hat{G}_{x_1, x_2}(f)$ due to limited observation time, where $\hat{G}_{x_1, x_2}(f)$ is an estimation of $G_{x_1, x_2}(f)$, so (2.7) should rewrite as

$$\hat{R}_{y_1, y_2}^{(g)}(\tau) = \int_{-\infty}^{\infty} \psi_g(f) \hat{G}_{x_1, x_2}(f) e^{j2\pi f\tau} df \quad (2.9)$$

Using (2.9), we could estimate the delay of the microphone pair. Different choices of $\psi_g(f)$ will affect the delay estimation. Carter [5] propose the method named PHAT (phase transform), which is

$$\psi_g(f) = \frac{1}{|G_{x_1, x_2}(f)|} \quad (2.10)$$

This method is very effective when the noise distributions of the two microphones are uncorrelated.

2.2.2 ES-GCC, a combination of TDE and Eigenspace Method

ES(eigen space)-GCC [8] uses the characteristic of Eigenspace Method to divide the signal into signal subspace and noise subspace. The signal subspace is the principle distribution of the recorded signal. Extracting the signal subspace would suppress the noise interference. Then, calculate the GCC focusing on the signal subspace.

Suppose there is a microphone array with M microphones, and there are d sources in the environment, the signal received by the m^{th} microphone is

$$x_m(t) = \sum_{k=1}^d a_{mk} s_k(t - \tau_{mk}) + n_m(t) \quad (2.11)$$

where a_{mk} is the gain from the k^{th} source to the m^{th} microphone. $n_m(t)$ is the noise received by the m^{th} microphone. Take the Fourier transform to (2.11) is

$$X_m(\omega_f, n) = \sum_{k=1}^d a_{mk} S_k(\omega_f, n) e^{-j\omega_f \tau_{mk}} + N_m(\omega_f, n) \quad (2.12)$$

$f = 1, 2, \dots, F$

where ω_f is the observed frequency band, and n is the index of temporal frame.

Rewrite (2.12) to matrix form

$$\mathbf{X}(\omega_f, n) = \mathbf{A}(\omega_f) \mathbf{S}(\omega_f, n) + \mathbf{N}(\omega_f, n) \quad (2.13)$$

where

$$\mathbf{X}^T(\omega_f, n) = [X_1(\omega_f, n), \dots, X_M(\omega_f, n)] \quad (2.14)$$

$$\mathbf{N}^T(\omega_f, n) = [N_1(\omega_f, n), \dots, N_M(\omega_f, n)] \quad (2.15)$$

$$\mathbf{S}^T(\omega_f, n) = [S_1(\omega_f, n), \dots, S_d(\omega_f, n)] \quad (2.16)$$

$$\mathbf{A}(\omega_f) = \begin{bmatrix} a_{11} e^{-j\omega_f \tau_{11}} & \dots & a_{1d} e^{-j\omega_f \tau_{1d}} \\ \vdots & \ddots & \vdots \\ a_{M1} e^{-j\omega_f \tau_{M1}} & \dots & a_{Md} e^{-j\omega_f \tau_{Md}} \end{bmatrix} \quad (2.17)$$

Now we calculate the correlation matrix and take the eigenvalue decomposition of it

$$\begin{aligned} \mathbf{R}_{xx}(\omega_f) &= \frac{1}{N} \sum_{n=1}^N \mathbf{X}(\omega_f, n) \mathbf{X}^H(\omega_f, n) \\ &= \sum_{i=1}^M \lambda_i(\omega_f) \mathbf{V}_i(\omega_f) \mathbf{V}_i^H(\omega_f) \end{aligned} \quad (2.18)$$

N is the total signal frame number used to estimate the correlation matrix.

$\lambda_i(\omega_f)$ is the eigenvalue and $\mathbf{V}_i(\omega_f)$ is the corresponding eigenvector, where

$$\lambda_1(\omega_f) \geq \lambda_2(\omega_f) \geq \dots \geq \lambda_M(\omega_f) \quad (2.19)$$

The MUSIC (multiple signals classification method) algorithm [4] divides the eigenvectors in (2.18) into two groups

1. $\mathbf{V}_1(\omega_f), \mathbf{V}_2(\omega_f) \cdots \mathbf{V}_d(\omega_f)$ is called the eigenvectors of the signal and

$\text{span}\{\mathbf{V}_1(\omega_f), \mathbf{V}_2(\omega_f) \cdots \mathbf{V}_d(\omega_f)\}$ is the signal subspace.

2. $\mathbf{V}_{d+1}(\omega_f), \mathbf{V}_{d+2}(\omega_f) \cdots \mathbf{V}_M(\omega_f)$ is called the eigenvectors of the noise and

$\text{span}\{\mathbf{V}_{d+1}(\omega_f), \mathbf{V}_{d+2}(\omega_f) \cdots \mathbf{V}_M(\omega_f)\}$ is the noise subspace.

The array manifold vector is formed according to the array geometry and plane wave assumption of the sound signal. Take uniform linear array as the example,

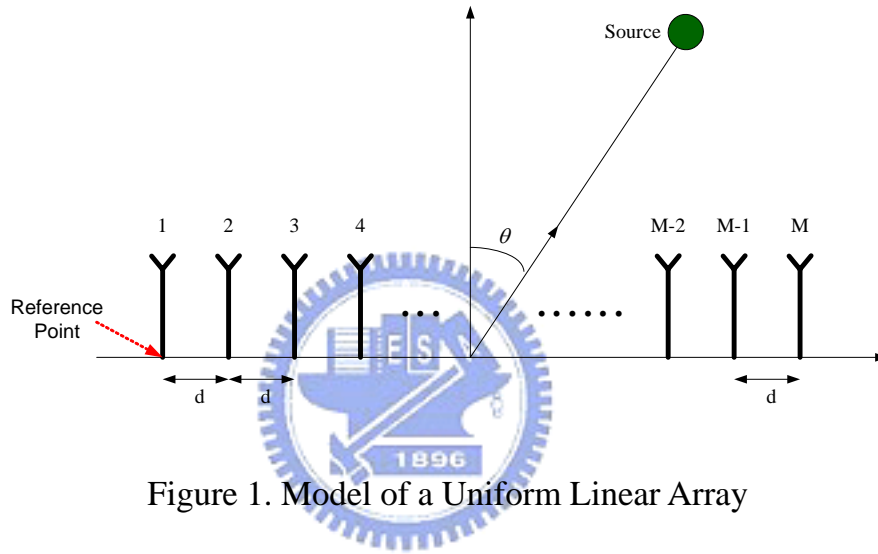


Figure 1. Model of a Uniform Linear Array

Set the first microphone as the reference point, the array manifold vector of the linear array is defined as

$$\mathbf{a}^T(\theta) = [1 \quad e^{j k_c \cdot d \cdot \sin \theta} \quad \dots \quad e^{j k_c \cdot d \cdot (M-1) \cdot \sin \theta}] \quad (2.20)$$

where $k_c = \frac{2\pi}{\lambda_c}$ indicates the number of wave fronts, λ_c is the wave length, d is

the distance between adjacent microphones, and M is the number of microphones.

The i^{th} element of the array manifold vector is the phase difference between the first microphone and the i^{th} microphone when the source direction is θ . The phase

difference in frequency domain is the temporal difference in time domain. Transform

the array manifold vector to time domain will be

$$\mathbf{a}^T_{\text{time-domain}}(\theta) = [1 \quad \frac{d}{\lambda_c} \sin \theta \quad \cdots \quad \frac{d}{\lambda_c} \cdot (m-1) \cdot \sin \theta] \quad (2.21)$$

MUSIC calculates the inner product of the array manifold vector and $\mathbf{V}_{d+1}(\omega_f), \mathbf{V}_{d+2}(\omega_f) \cdots \mathbf{V}_M(\omega_f)$ to determine the direction of the source.

$$\theta_{DOA} = \max \left(\frac{1}{\mathbf{a}^H(\theta) \cdot \left(\sum_{i=d+1}^M \mathbf{V}_i \cdot \mathbf{V}_i^H \right) \cdot \mathbf{a}(\theta)} \right) \quad (2.22)$$

The signal subspace is orthogonal to noise subspace, so (2.20) could rewrite as

$$\theta_{DOA} = \max \left(\mathbf{a}^H(\theta) \left(\sum_{i=1}^d \mathbf{V}_i \cdot \mathbf{V}_i^H \right) \mathbf{a}(\theta) \right) \quad (2.23)$$

The array manifold vector represents the phase difference between microphones. Since MUSIC measures the projection of the array manifold vector to the signal subspace, the principle eigenvector of the signal subspace should contains the phase relation between microphones. So, we extract $\mathbf{V}_1(\omega_f)$ from the signal subspace. $\mathbf{V}_1(\omega_f)$ is the principle axis of the microphone array at frequency ω_f , and it could be expressed as

$$\mathbf{V}_1(\omega_f) = [V_{11}(\omega_f) \quad V_{12}(\omega_f) \quad \cdots \quad V_{1M}(\omega_f)]^T \quad (2.24)$$

The principle matrix for all frequency is

$$\mathbf{E}_1 = \begin{bmatrix} V_{11}(\omega_1) & V_{11}(\omega_2) & \cdots & V_{11}(\omega_F) \\ V_{12}(\omega_1) & V_{12}(\omega_2) & \cdots & V_{12}(\omega_F) \\ \vdots & \vdots & \ddots & \vdots \\ V_{1M}(\omega_1) & V_{1M}(\omega_2) & \cdots & V_{1M}(\omega_F) \end{bmatrix} \quad (2.25)$$

In (2.25), the i^{th} column is the principle vector at sound frequency ω_i . We estimate the phase relation between microphones by taking GCC to \mathbf{E}_1 , so the

ES-GCC between the i^{th} microphone and the j^{th} is defined as

$$\mathbf{R}_{x_i x_j}(\tau) = \int_{\omega_1}^{\omega_F} V_{1i}(\omega) V_{1j}^*(\omega) e^{j\omega\tau} d\omega \quad (2.26)$$

And the estimation of the time delay between microphones is

$$\hat{\tau}_{ES-GCC} = \arg \max_{\tau} \mathbf{R}_{x_i x_j}(\tau) \quad (2.27)$$

2.3 Performance of Different Eigenspace Methods in Real Time

Application

In this section we'll discuss the functionality of ES-GCC and MUSIC. The time frame number required for ES-GCC to calculate a DOA is $1500 \times (200-1) + 2560$ samples, which is 18.82 sec. The MUSIC algorithm needs 2560×30 samples, which is 4.80 sec. So, ES-GCC is more time consuming. However, ES-GCC is able to calculate multiple sound sources without previous knowledge, while MUSIC is only able to estimate single source without other information. Another aspect is that ES-GCC requires a TDE step before estimating the direction, and MUSIC estimate the DOA directly. Since there might be TDE error in ES-GCC which causes more serious TDA estimation error, MUSIC is the method that has less estimation error. Finally, in the MUSIC algorithm the array manifold vector, which was determined by the array geometry, may not be a direct mapping with the source direction, so it's computation consuming. On the other hand, ES-GCC only examines the delay relation between microphone pairs. It costs less computation to derive the DOA of the source.

Chapter 3. BEARINGS-ONLY SLAM ALGORITHM

3.1 Between DOA estimation and Bearings-Only SLAM

The connection between DOA estimation algorithm and Bearings Only SLAM algorithm is essential. The DOA algorithm estimates the sound emitting direction of multiple sources. The output information may not be of the same index order, and the total number of estimated sound source is not static. The later algorithm should be able to handle incomplete measurement and unknown data association.

The Bearings-Only SLAM is a probabilistic robotics algorithm. It uses bearing information of landmarks in the environment to simultaneously localize the robot in the environment and to realize the location of the landmarks. The uncertain information from the output of DOA estimation algorithm is handled as the measurement input. The data association is not necessary to be known because the localization result based on wrong association is going to be eliminated using this algorithm. The remaining detail of Bearings-Only SLAM is introduced in the following section of this chapter.

3.2 Introduction of Probabilistic Robotics

Probabilistic robotics [6] is alternative to the conventional deterministic robotic. The key idea in probabilistic robotics is to represent uncertainty explicitly using the calculus of probability theory. Instead of relying on a single result as to what might be the case, probabilistic algorithm represents information by probability distributions over a whole space of guesses. They can represent ambiguity and degree of belief in a mathematically sound way. In contrast with traditional programming techniques in

robotics, probabilistic approaches tend to be more robust in the face of sensor limitations and model limitation. This enables them to scale much better to complex real-world environments than previous diagram, where uncertainty is of greater importance. Probabilistic algorithms are the only known working solutions to robotic estimation problems such as localization.

3.3 Bayes Filter

3.3.1 State Estimation using Probabilistic Generative Laws

Environments are characterized by *state*. The state is a collection of all information of the robot and its environment that can impact the future. It includes variables regarding to the robot itself, such as its pose, velocity, whether or not its sensors are functioning correctly, and so on. The robot uses its sensors to obtain information about the state of the environment, and the result of such perceptual interaction will be called *measurement* (or *observation*). The evolution of state and measurements is governed by probabilistic laws. Mathematically, the emergence of state x_t is conditioned on all past states, measurement, and control inputs, so the probability distribution of x_t could be expressed in the conditional probability form

$$p(x_t | x_{0:t-1}, z_{1:t-1}, u_{1:t}) \quad (3.1)$$

where $x_{0:t-1}$ are the past states from time index 0 to $t-1$, $z_{1:t-1}$ are the past measurements, and $u_{1:t}$ are the past control input. Note that the control input u_1 is executed first, and then we take the measurement z_1 . To simplify the expression, a state x_t should be a sufficient summary of all that happened in previous time steps. In particular, x_{t-1} is a sufficient statistic of all previous controls and measurements up to

this point of time, that is, $u_{1:t-1}$ and $z_{1:t-1}$. So, if we already knew state x_{t-1} , only the control input u_t will affect the present state. In mathematical expression

$$p(x_t | x_{0:t-1}, z_{1:t-1}, u_{1:t}) = p(x_t | x_{t-1}, u_t) \quad (3.2)$$

This property is called conditional independence.

Another key concept in the probabilistic robotics is *belief*. The belief is the robot's internal knowledge about the state of the environment. Probabilistic robotics represents the beliefs through conditional probability distributions. Belief distributions are posterior probabilities over state variables conditioned on the available data. We denote belief over a state variable x_t by $bel(x_t)$

$$bel(x_t) = p(x_t | z_{1:t}, u_{1:t}) \quad (3.3)$$

where the posterior is the probability distribution over the state x_t at time t , conditioned on all past measurements $z_{1:t}$ and all past control $u_{1:t}$. Notice that the belief is assumed to be taken after the measurement z_t . If we take the belief before the measurement, which is usually the case for real application, the posterior will be

$$\overline{bel}(x_t) = p(x_t | z_{1:t-1}, u_{1:t}) \quad (3.4)$$

This probability distribution is often referred to as *prediction*. The Bayes filter predicts the posterior $\overline{bel}(x_t)$ based on the previous state posterior $bel(x_{t-1})$, and then incorporates $\overline{bel}(x_t)$ with the measurement at time t . The corporation is called *measurement update*.

```

1. Algorithm Bayes_filter( $bel(x_{t-1}), u_t, z_t$ ):
2.   for all  $x_t$  do
3.      $\overline{bel}(x_t) = \int p(x_t | u_t, x_{t-1}) bel(x_{t-1}) dx_{t-1}$  (prediction)
4.      $bel(x_t) = \eta p(z_t | x_t) \overline{bel}(x_t)$  (update)
5.   end for
6.   return  $bel(x_t)$ 

```

Figure 2. The Pseudo-Algorithm for Bayes Filtering

As in Figure 2, the Bayes filter algorithm possesses two essential steps. In line 3, it processes the control u_t . It does so by calculating a belief over the state x_t based on the prior belief over state x_{t-1} and the control u_t . In particular, the belief $\overline{bel}(x_t)$ that the robot assigns to state x_t is obtained by the integral (sum) of the product of two distributions: the prior assigned to x_{t-1} , and the probability that control u_t induces a transition from x_{t-1} to x_t . This step is call predict.

The second step is update. In line 4, the Bayes filter algorithm multiplies the belief $\overline{bel}(x_t)$ by the probability that the measurement z_t may have been observed. The result is normalized using η .

3.3.2 Parametric Filter – Kalman Filter and Extended Kalman Filter

The Kalman filter was invented by Swerling and Kalman [7] as a technique for filtering and prediction in *linear Gaussian systems*. The Kalman filter represents beliefs by the mean μ_t and covariance Σ_t , and the posteriors are Gaussian. The Kalman filter has several properties

1. The state transition probability $p(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_t)$ is a linear function with added Gaussian noise. Which is

$$\mathbf{x}_t = \mathbf{A}_t \mathbf{x}_{t-1} + \mathbf{B}_t \mathbf{u}_t + \boldsymbol{\varepsilon}_t \quad (3.5)$$

Here \mathbf{x}_t and \mathbf{x}_{t-1} are state vectors, and \mathbf{u}_t is the control vector at time t .

The term $\boldsymbol{\varepsilon}_t$ is a Gaussian random vector that models the uncertainty generated by the state transition. Its mean is zero vector and covariance is \mathbf{R}_t .

So the total probability distribution is

$$\begin{aligned} & p(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_t) \\ &= \frac{1}{\sqrt{|2\pi \cdot \mathbf{R}_t|}} \exp\left\{-\frac{1}{2}(\mathbf{x}_t - \mathbf{A}_t \mathbf{x}_{t-1} - \mathbf{B}_t \mathbf{u}_t)^T \mathbf{R}_t^{-1} (\mathbf{x}_t - \mathbf{A}_t \mathbf{x}_{t-1} - \mathbf{B}_t \mathbf{u}_t)\right\} \end{aligned} \quad (3.6)$$

2. The measurement probability $p(\mathbf{z}_t | \mathbf{x}_t)$ is also a linear in the argument transition.

$$\mathbf{z}_t = \mathbf{C}_t \mathbf{x}_t + \boldsymbol{\delta}_t \quad (3.7)$$

The probability distribution of $p(\mathbf{z}_t | \mathbf{x}_t)$ should be

$$\begin{aligned} & p(\mathbf{z}_t | \mathbf{x}_t) \\ &= \frac{1}{\sqrt{|2\pi \cdot \mathbf{Q}_t|}} \exp\left\{-\frac{1}{2}(\mathbf{z}_t - \mathbf{C}_t \mathbf{x}_t)^T \mathbf{Q}_t^{-1} (\mathbf{z}_t - \mathbf{C}_t \mathbf{x}_t)\right\} \end{aligned} \quad (3.8)$$

where \mathbf{Q}_t is the covariance of the zero-mean Gaussian random vector $\boldsymbol{\delta}_t$,

3. The initial probability distribution of $p(\mathbf{x}_0)$ is a Gaussian distribution. In this case the propagation of $p(\mathbf{x}_t)$ is guaranteed to be Gaussian distribution.

The above three characteristics are sufficient for the use of Kalman filter. Figure 3 is the detail algorithm procedure.

1. **Algorithm Kalman_filter**($\boldsymbol{\mu}_{t-1}, \boldsymbol{\Sigma}_{t-1}, \mathbf{u}_t, \mathbf{z}_t$):
2. $\bar{\boldsymbol{\mu}}_t = \mathbf{A}_t \boldsymbol{\mu}_{t-1} + \mathbf{B}_t \mathbf{u}_t$ (prediction)
3. $\bar{\boldsymbol{\Sigma}}_t = \mathbf{A}_t \boldsymbol{\Sigma}_{t-1} \mathbf{A}_t^T + \mathbf{R}_t$ (prediction)
4. $\mathbf{K}_t = \bar{\boldsymbol{\Sigma}}_t \mathbf{C}_t^T (\mathbf{C}_t \bar{\boldsymbol{\Sigma}}_t \mathbf{C}_t^T + \mathbf{Q}_t)^{-1}$
5. $\boldsymbol{\mu}_t = \bar{\boldsymbol{\mu}}_t + \mathbf{K}_t (\mathbf{z}_t - \mathbf{C}_t \bar{\boldsymbol{\mu}}_t)$ (update)
6. $\boldsymbol{\Sigma}_t = (\mathbf{I} - \mathbf{K}_t \mathbf{C}_t) \bar{\boldsymbol{\Sigma}}_t$ (update)
7. return $\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t$

Figure 3. The Kalman Filter Algorithm for
Linear Gaussian state transitions and measurements

After knowing the iterative procedure of Kalman filter, we now examine another version of Kalman filter, which is called EKF (extended Kalman filter). According to (3.5) and (3.7), the observations are linear functions of the state and the state transition is also linear function. This assumption is important to Kalman filter but not adequate for real world. The key point is that generally the transition and observation procedure of state are nonlinear functions. In mathematical form,

$$\mathbf{x}_t = \mathbf{g}(\mathbf{x}_{t-1}, \mathbf{u}_t) + \boldsymbol{\varepsilon}_t \quad (3.9)$$

$$\mathbf{z}_t = \mathbf{h}(\mathbf{x}_t) + \boldsymbol{\delta}_t \quad (3.10)$$

Since the assumption we made about linear transition no longer exist, there should be some approximation about the nonlinear function. The most general terminology is linearization via first order Taylor expansion. Taylor expansion constructs a linear approximation to a function g from g 's value and slope near the function point. The slope is given by the partial derivative

$$\mathbf{g}'(\mathbf{u}_t, \mathbf{x}_{t-1}) := \frac{\partial \mathbf{g}(\mathbf{u}_t, \mathbf{x}_{t-1})}{\partial \mathbf{x}_{t-1}} \quad (3.11)$$

From the expression above, \mathbf{g} is approximated by its value at μ_{t-1} and u_t , so

$$\begin{aligned} \mathbf{g}(\mathbf{u}_t, \mathbf{x}_{t-1}) &\approx \mathbf{g}(\mathbf{u}_t, \boldsymbol{\mu}_{t-1}) + \mathbf{g}'(\mathbf{u}_t, \boldsymbol{\mu}_{t-1}) \cdot (\mathbf{x}_{t-1} - \boldsymbol{\mu}_{t-1}) \\ &= \mathbf{g}(\mathbf{u}_t, \boldsymbol{\mu}_{t-1}) + \mathbf{G}_t \cdot (\mathbf{x}_{t-1} - \boldsymbol{\mu}_{t-1}) \end{aligned} \quad (3.12)$$

The matrix \mathbf{G}_t is defined as the Jacobian matrix, and it should be a function of various value of μ_{t-1} and u_t . In the form of a Gaussian distribution, the state transition probability distribution approximation is

$$\begin{aligned} p(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_t) &= \frac{1}{\sqrt{|2\pi \cdot \mathbf{R}_t|}} \exp\left\{-\frac{1}{2}[\mathbf{x}_t - \mathbf{g}(\mathbf{u}_t, \boldsymbol{\mu}_{t-1}) - \mathbf{G}_t \cdot (\mathbf{x}_{t-1} - \boldsymbol{\mu}_{t-1})]^T \cdot \right. \\ &\quad \left. \mathbf{R}_t^{-1}[\mathbf{x}_t - \mathbf{g}(\mathbf{u}_t, \boldsymbol{\mu}_{t-1}) - \mathbf{G}_t \cdot (\mathbf{x}_{t-1} - \boldsymbol{\mu}_{t-1})]\right\} \end{aligned} \quad (3.13)$$

EKF implements the same linearization for the measurement function h . Here the Taylor expansion is developed around $\bar{\boldsymbol{\mu}}_t$

$$\begin{aligned} h(\mathbf{x}_t) &\approx h(\bar{\boldsymbol{\mu}}_t) + h'(\bar{\boldsymbol{\mu}}_t) \cdot (\mathbf{x}_t - \bar{\boldsymbol{\mu}}_t) \\ &= h(\bar{\boldsymbol{\mu}}_t) + \mathbf{H}_t \cdot (\mathbf{x}_t - \bar{\boldsymbol{\mu}}_t) \end{aligned} \quad (3.14)$$

The Gaussian form would be

$$\begin{aligned} p(\mathbf{z}_t | \mathbf{x}_t) &= \frac{1}{\sqrt{|2\pi \cdot \mathbf{Q}_t|}} \exp\left\{-\frac{1}{2}[\mathbf{z}_t - h(\bar{\boldsymbol{\mu}}_t) - \mathbf{H}_t(\mathbf{x}_t - \bar{\boldsymbol{\mu}}_t)]^T \right. \\ &\quad \left. \mathbf{Q}_t^{-1}[\mathbf{z}_t - h(\bar{\boldsymbol{\mu}}_t) - \mathbf{H}_t(\mathbf{x}_t - \bar{\boldsymbol{\mu}}_t)]\right\} \end{aligned} \quad (3.15)$$

And the total procedure of EKF is in figure 4.

1. **Algorithm Extended_Kalman_filter**($\boldsymbol{\mu}_{t-1}, \boldsymbol{\Sigma}_{t-1}, \mathbf{u}_t, \mathbf{z}_t$):
2. $\bar{\boldsymbol{\mu}}_t = \mathbf{g}(\boldsymbol{\mu}_{t-1}, \mathbf{u}_t)$ (prediction)
3. $\bar{\boldsymbol{\Sigma}}_t = \mathbf{G}_t \boldsymbol{\Sigma}_{t-1} \mathbf{G}_t^T + \mathbf{R}_t$ (prediction)
4. $\mathbf{K}_t = \bar{\boldsymbol{\Sigma}}_t \mathbf{H}_t^T (\mathbf{H}_t \bar{\boldsymbol{\Sigma}}_t \mathbf{H}_t^T + \mathbf{Q}_t)^{-1}$
5. $\boldsymbol{\mu}_t = \bar{\boldsymbol{\mu}}_t + \mathbf{K}_t (\mathbf{z}_t - \mathbf{h}(\bar{\boldsymbol{\mu}}_t))$ (update)
6. $\boldsymbol{\Sigma}_t = (\mathbf{I} - \mathbf{K}_t \mathbf{H}_t) \bar{\boldsymbol{\Sigma}}_t$ (update)
7. return $\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t$

Figure 4. The Extended Kalman Filter Algorithm

Due to the simplicity and its computational efficiency, the EKF is almost the most frequently used state estimation terminology in robotics. Each update requires time $O(k^{2.4} + n^2)$, where k is dimension of the measurement vector \mathbf{z}_t , and $\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t$ is the dimension of the state vector \mathbf{x}_t . Other algorithms, such as the particle filter, may require time exponential in n

3.3.3 Nonparametric Filter - Particle Filter

Particle filter does not rely on a fixed functional form of the posterior, like Gaussian distribution in Kalman filter. It approximates posteriors by a finite number of values, each roughly corresponding to a region in state space. The quality of the approximation depends on the number of parameters used to represent the posterior. As the number of parameters goes to infinity, the method tends to converge uniformly to the correct posterior. The key idea is to represent the posterior $bel(x_t)$ by a set of random state samples drawn from this posterior. In particle filters, the samples of a posterior distribution are called particles.

$$\boldsymbol{\chi}_t := \mathbf{x}_t^{[1]}, \mathbf{x}_t^{[2]}, \dots, \mathbf{x}_t^{[M]} \quad (3.16)$$

Each particle $\mathbf{x}_t^{[m]}$ is a representation of the state at time t . In other words, a particle is a hypothesis as to what the true world state may be at time t . M is the number of particles in the particle set $\boldsymbol{\chi}_t$. Ideally, the likelihood for a state to be included in the particle set $\boldsymbol{\chi}_t$ shall be proportional to its posterior $bel(x_t)$

$$\mathbf{x}_t^{[m]} \sim p(\mathbf{x}_t | \mathbf{z}_{1:t}, \mathbf{u}_{1:t}) \quad (3.17)$$

Just like all other Bayes filter algorithms discussed, the particle filter algorithm constructs the belief $bel(x_t)$ recursively from the $bel(x_{t-1})$ one time step earlier, which means the particle set $\boldsymbol{\chi}_t$ recursively from the set $\boldsymbol{\chi}_{t-1}$. The input of particle filter is the particle set $\boldsymbol{\chi}_{t-1}$, along with the most recent control \mathbf{u}_t and the most recent measurement \mathbf{z}_t . To get the predicted probability distribution $\overline{bel}(x_t)$, we sample the new particles $\mathbf{x}_t^{[m]}$ according to the distribution of the previous step $p(\mathbf{x}_t | \mathbf{u}_t, \mathbf{x}_{t-1})$. To incorporate the measurement \mathbf{z}_t into the particle set, we calculate the *importance factor* $w_t^{[m]}$, which are obtained according to the probability of the measurement \mathbf{z}_t under particle $\mathbf{x}_t^{[m]}$, given by $w_t^{[m]} = p(\mathbf{z}_t | \mathbf{x}_t^{[m]})$. If we interpret $w_t^{[m]}$ as the weight of a particle, the set of weighted particles represent approximately the Bayes filter posterior $bel(x_t)$. The last step is called *resample*. Since each of the particles already has a corresponding weight, the weigh value could be interpreted as the probability that this particle would be chosen again. Resampling algorithm transforms a particle set of M particles into another particle set of the same size. Before the resampling step, they were distributed according to $\overline{bel}(x_t)$, after the

resampling the particles distributed according to the posterior

$bel(x_t) = \eta p(\mathbf{z}_t | \mathbf{x}_t^{[m]}) \overline{bel}(x_t)$. The illustrated algorithm flow is in Figure 5.

```

1. Algorithm Particle_filter( $\chi_{t-1}, \mathbf{u}_t, \mathbf{z}_t$ ):
2.    $\overline{\chi}_t = \chi_t = \mathbf{0}$ 
3.   for  $m=1$  to  $M$  do(prediction)
4.     sample  $\mathbf{x}_t^{[m]} \sim p(\mathbf{x}_t | \mathbf{u}_t, \mathbf{x}_{t-1}^{[m]})$ 
5.      $w_t^{[m]} = p(\mathbf{z}_t | \mathbf{x}_t^{[m]})$ 
6.      $\overline{\chi}_t = \overline{\chi}_t + [\mathbf{x}_t^{[m]}, w_t^{[m]}]$ 
7.   end for
8.   for  $m=1$  to  $M$  do
9.     draw  $i$  with probability  $\propto w_t^{[i]}$ 
10.    add  $\mathbf{x}_t^{[i]}$  to  $\chi_t$ 
11.  end for
12. return  $\chi_t$ 

```

Figure 5. The Particle Filter Algorithm, a Bayes Filter Based on Importance Sampling

3.4 SLAM

3.4.1 Problem Definition

The *simultaneous localization and mapping* problem is commonly abbreviated as *SLAM*, and is also known as Concurrent Mapping and Localization. SLAM problem arise when the robot does not have access to a map of the environment, nor does it know its own pose. All it is given are measurement $z_{1:t}$ and controls $u_{1:t}$. In SLAM, the robot acquires a map of its environment while simultaneously localizing itself

relative to this map. From a probabilistic perspective, there are two main forms of the SLAM problem, which are both of equal practical importance. The *online SLAM* estimates the posterior over the momentary pose along with the map.

$$p(\mathbf{x}_t, \mathbf{m} \mid \mathbf{z}_{1:t}, \mathbf{u}_{1:t}) \quad (3.18)$$

where \mathbf{x}_t is the pose at time t , \mathbf{m} is the map. This problem is called online SLAM problem since it only involves the estimation of variables that persist at time t . The other category is called the *full SLAM*. In full SLAM, we calculate a posterior over the entire path $\mathbf{x}_{1:t}$ along with the map, instead of just the current pose \mathbf{x}_t .

$$p(\mathbf{x}_{1:t}, \mathbf{m} \mid \mathbf{z}_{1:t}, \mathbf{u}_{1:t}) \quad (3.19)$$

The mathematical relation between (3.18) and (3.19) is shown below.

$$p(\mathbf{x}_t, \mathbf{m} \mid \mathbf{z}_{1:t}, \mathbf{u}_{1:t}) = \int \int \cdots \int p(\mathbf{x}_{1:t}, \mathbf{m} \mid \mathbf{z}_{1:t}, \mathbf{u}_{1:t}) d\mathbf{x}_1 d\mathbf{x}_2 \cdots d\mathbf{x}_{t-1} \quad (3.20)$$

In practice, calculating a full posterior like (3.19) is usually infeasible. Problems arise from the high dimensionality of the continuous parameter space, and the large number of discrete correspondence variables. Many state-of-the-art SLAM algorithms construct maps with tens of thousands of features, or more. Even under known correspondence, the posterior over those maps alone involves probability distributions over space with 105 or more dimensions. This is opposite to localization problems, in which posteriors were estimated over three-dimensional continuous spaces. Not to say in most applications the correspondences are unknown. The number of possible assignments to the vector of all correspondence variables grows exponentially. Thus practical SLAM algorithms that can cope with the correspondence problem must rely on approximations.

3.4.2 FastSLAM

FastSLAM is a combinational algorithm that is using both the concept of particle filter and the extended Kalman filter. Particle filters are at the core of some of the most effective robotics algorithm. However the particle filter is not so applicable to the SLAM algorithm due to the curse of dimensionality, it scales exponentially with the number of dimensions of the estimation problem. A straightforward implementation of particle filters for the SLAM problem would fail because of the large number of variables involved on describing a map.

In a SLAM problem with known correspondence, there is a conditional independence between any two disjoint set of features in the map, given the robot pose. So we could estimate the location of all features independently of each other. Dependencies on these estimates arise only through robot pose uncertainty. This structural characteristic makes it possible to apply *Rao-Blackwellized Particle Filter* (RB particle filter) to SLAM problem. RB particle filter uses particles to represent the posterior over some variables, along with parametric PDF to represent all other variables.

We use particle filter to estimate the robot path. Since the conditional independent characteristic holds, the mapping problem can be factored into many separate problems, one for each feature in the map. Each single map feature is estimated using a low-dimensional EKF. This is different from other SLAM algorithms that use single high-dimensional Gaussian to estimate the all features jointly.

The advantage of FastSLAM is that it could be implemented in time logarithmic in the number of features, so it's computational efficient. Another key advantage of

FastSLAM is that the data association decisions can be made on per-particle basis. The ability to pursue multiple data associations simultaneously makes FastSLAM significantly more robust to data association problems than algorithms based on incremental maximum likelihood data association.

	robot path	feature 1	feature 2	...	feature N
Particle $k = 1$	$\mathbf{x}_{1:t}^{[1]} = \{(x, y, \theta)^T\}_{1:t}^{[1]}$	$\mu_1^{[1]}, \Sigma_1^{[1]}$	$\mu_2^{[1]}, \Sigma_2^{[1]}$...	$\mu_N^{[1]}, \Sigma_N^{[1]}$
Particle $k = 2$	$\mathbf{x}_{1:t}^{[2]} = \{(x, y, \theta)^T\}_{1:t}^{[2]}$	$\mu_1^{[2]}, \Sigma_1^{[2]}$	$\mu_2^{[2]}, \Sigma_2^{[2]}$...	$\mu_N^{[2]}, \Sigma_N^{[2]}$
⋮					
Particle $k = M$	$\mathbf{x}_{1:t}^{[M]} = \{(x, y, \theta)^T\}_{1:t}^{[M]}$	$\mu_1^{[M]}, \Sigma_1^{[M]}$	$\mu_2^{[M]}, \Sigma_2^{[M]}$...	$\mu_N^{[M]}, \Sigma_N^{[M]}$

Table 1. Particles in FastSLAM are composed of a path estimate and

a set of estimators of individual feature locations with associated covariance

Particles in the basic FastSLAM algorithm are of the form shown in Table 1. Each particle contains an estimated robot pose, denoted $\mathbf{x}_{1:t}^{[k]}$, and a set of Kalman filters with mean $\mu_j^{[k]}$ and covariance $\Sigma_j^{[k]}$, one for each feature m_j in the map. Here $[k]$ is the index of the particle. As usual, the total number of particles is denoted M . The basic step of the FastSLAM includes several steps. At each time step t , retrieve a pose $\mathbf{x}_{t-1}^{[k]}$ from the particle set, sample a new pose according to the distribution $\mathbf{x}_t^{[k]} \sim p(\mathbf{x}_t | \mathbf{x}_{t-1}^{[k]}, \mathbf{u}_t)$. Then for each observed feature z_t^j , identify the correspondence j and update the corresponding EKF by updating mean $\mu_{j,t}^{[k]}$ and covariance $\Sigma_{j,t}^{[k]}$. Each new particle should have a new importance weight $w^{[k]}$. The final step is resample, which is sampling the M particles with the probability proportional to $w^{[k]}$

3.4.3 Unknown Data Association

The data association is also mentioned as *correspondence* in this thesis. In most of the Bayes filter process, the data association problem exists. Every time we take a measurement of the map, we don't necessarily know which feature the measurement belongs to (usually we don't have this piece of information). Generally the data association techniques are using argument such as maximum likelihood. They have only single data association per measurement for the entire filter, and once the association is incorrect, the update procedure fails and the filter diverges. The key advantage of using particle filters for SLAM is that each particle can rely on its own, local data association decision. Thus, the filter is actually sampling over possible data association decisions. Since there are multiple correspondence decisions, as long as a small subset of the particles is based on the correct data association, data association errors are not as fatal as in EKF approaches. Particles with wrong correspondence will possess inconsistent map, which decrease the weight of those particles, and hence they are more likely to be sampled out in the future resample step.

3.5 Simulation Result of Fast SLAM

In this simulation we'll examine the effectiveness of the FastSLAM algorithm. Assume there are 6 features in the environment that are to be tracked, and the robot will walk through a certain path that is previously given. The space is 700cm*700cm. Maximum detectable range of the bearing sensors is 300cm. The algorithm generates particles according to the very first bearing measurements. Since there is a maximum detectable range, we sample the initial particle ranges according to a uniform distribution of half of the maximum range. The procedure is as Figure 6.

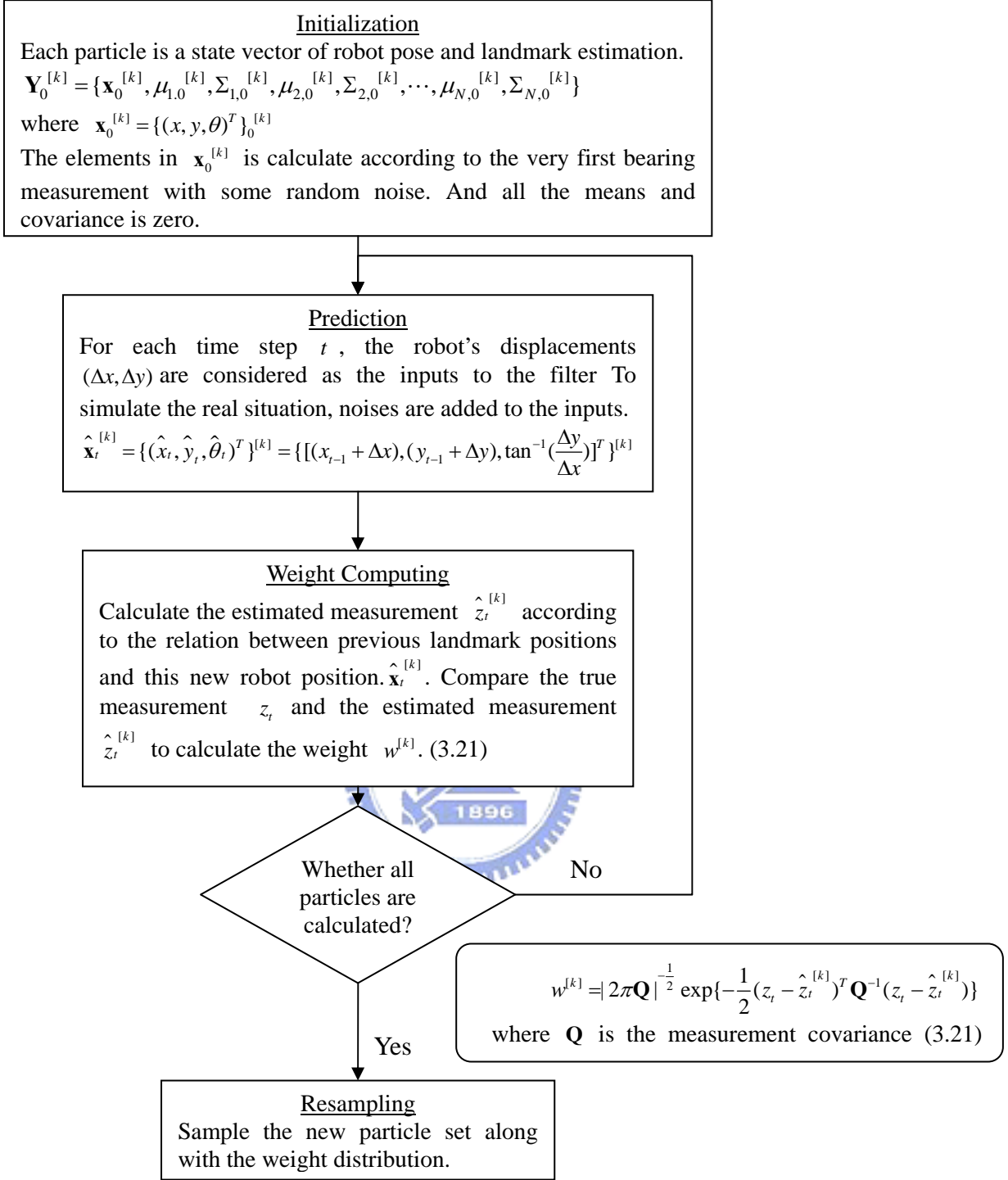


Figure 6. The Simulation Procedure of the FastSLAM Algorithm

Notice that in (3.21), the weight value will be small if the estimated measurement is different from the real measurement. And since there is resample step, we don't have to worry about data association problem.

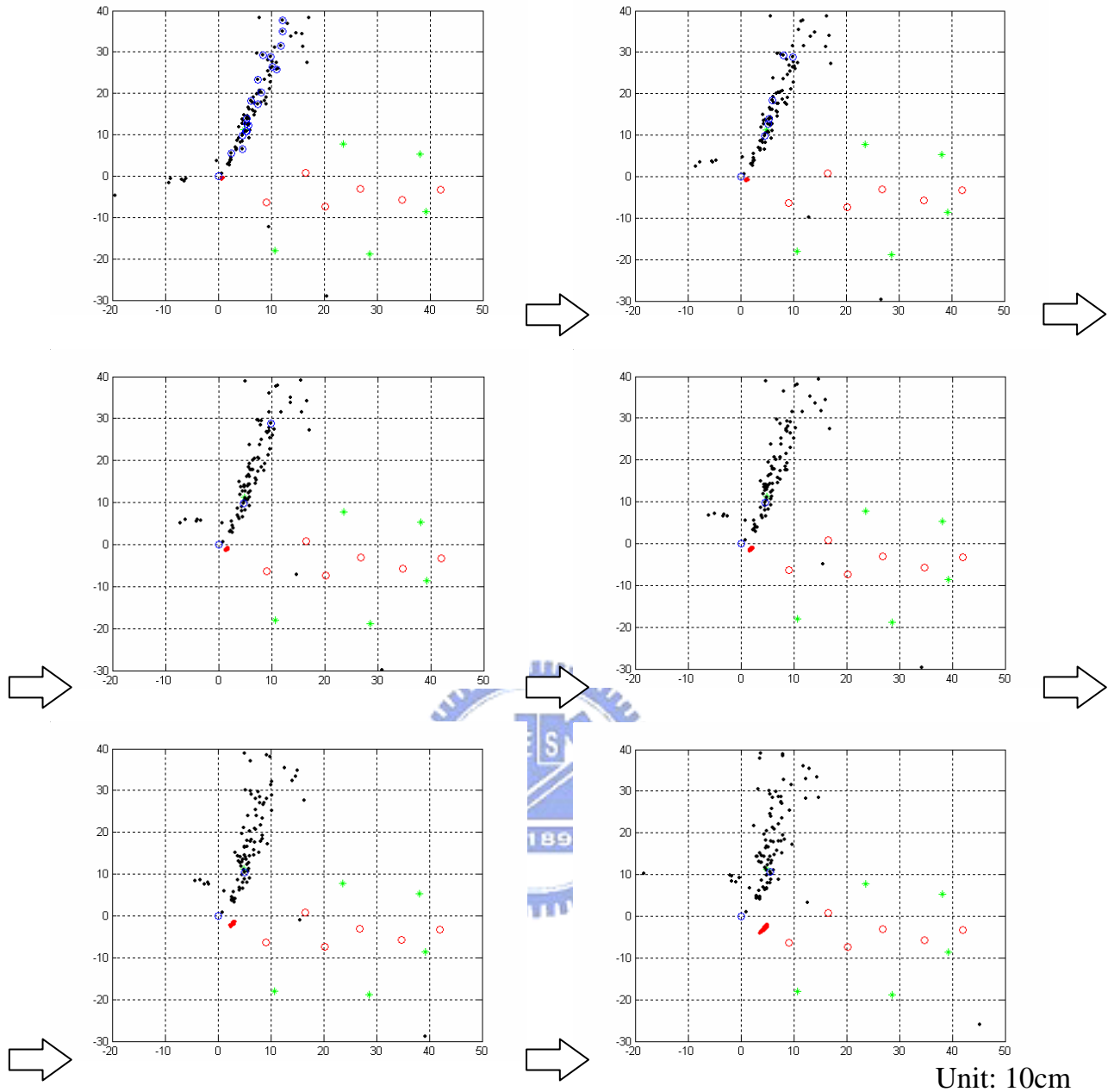


Figure 7. Simulation Results of the FastSLAM Algorithm without Resample
(Only single feature is illustrated)

Figure 7 shows a sequential graph of the simulation. The six green stars are the feature that to be localized. Red dots are the estimation of the current robot position. Red circles are the waypoint of the path. All the black particles are trying to localize the feature at $(4.95, 11.26)$. And the blue circles are the particles with larger weight value (there is also a blue circle at the origin). To explain what the figure is all about,

we conclude the following observation. Originally there are a lot of particles that have high weight, and as time step increase, the weight will concentrate to several mostly correct particle. Even if the resample state is not implemented, the outlier particles will converge to the feature's neighborhood.

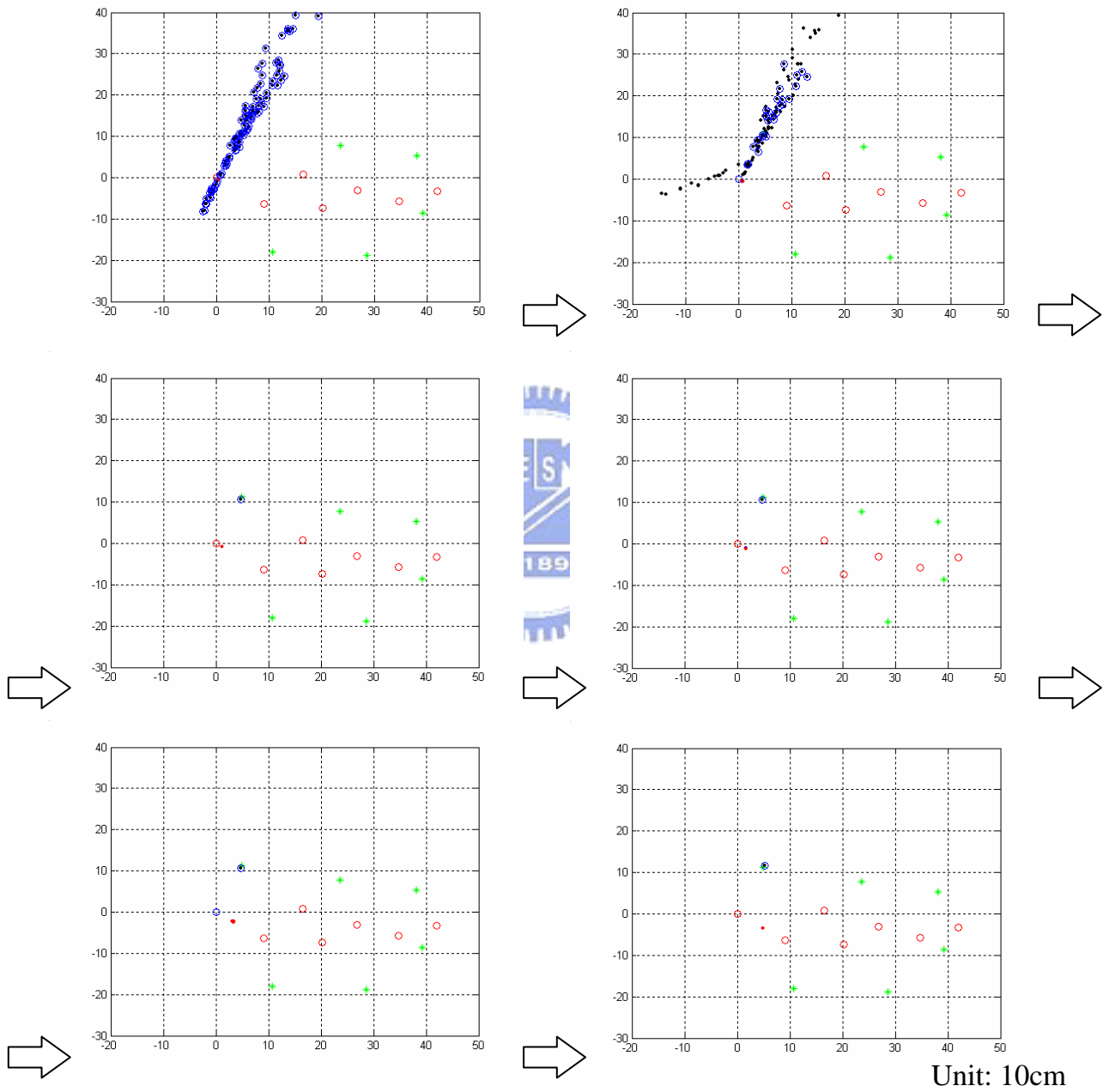


Figure 8. Simulation Results of the FastSLAM Algorithm with Resample

In Figure 8, it is shown that with the help of resample, all the low-weight particles will be sampled out. The filter converges in only few time steps.

Chapter 4. Experimental Result and Discussion

4.1 Introduction of the Robot Platform

In the experimental results presented afterward, we're using a research and development platform called *PIONEER 3-DX* from **MobileRobots Inc.** Some detail specification is written in Table 2.

Pioneer 3-DX Research Robot		
Physical	Length	44.5 cm
	Width	40.0 cm
	Height	24.5 cm
	Weight (with battery)	9 kg
Power	Battery	12V sealed, lead-acid
	Charge Time	12 hours
	Run time	18-24 hours
Mobility	Drive	2-wheel drive, plus rear balancing caster
	Gear ratio	38.3:1
	Pushing force	6 kg
	Swing radius	32 cm
	Translate speed max	1.2 m/sec
Standard position encoders		500 tick encoders
Communications ports		3 RS-232 serial ports on microcontroller
Main power switch		Robot power; 12VDC; red LED indicator

Table 2. The Specification of the *POINEER 3-DX* Robot [9]



Figure 9. Overall Picture of the Mobile Robot Platform

The main purpose of using this robot platform is to record the robot path using the included position encoders. These encoder data will be considered as the input to the FastSLAM algorithm. We use a laptop to control the robot through the RS-232 port. A wireless access point is mounted on the robot so that the instruction could be done remotely.

Another important hardware is the microphone array. Figure 10 shows the architecture of the array, which contains 8 digital microphones, an FPGA, and the USB communication module. The digital microphones are using the MEMS technology to embed sigma-delta modulators in the microphones. Digital microphones have the advantages of little signal crosstalk, low circuit noise, and small circuit size due to needless of A/D converter. The FPGA will take decimation to the microphone input data. It passes the 1.2MHz digital microphone signal through a second-order low pass filter (LPF), and down samples the 1.2 MHz 75 times to get a 16 KHz sound signal. Afterward, the USB will select the channel number through a MUX, and simultaneously transmit the 8-channel 16 KHz data to the laptop.

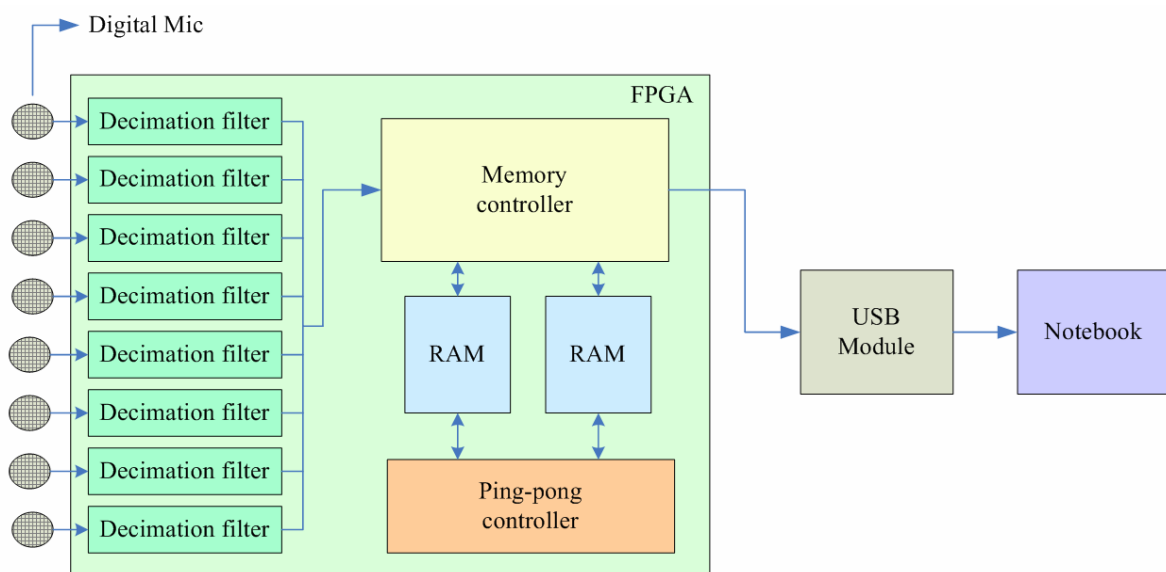


Figure 10. Block Diagram of the Microphone Array

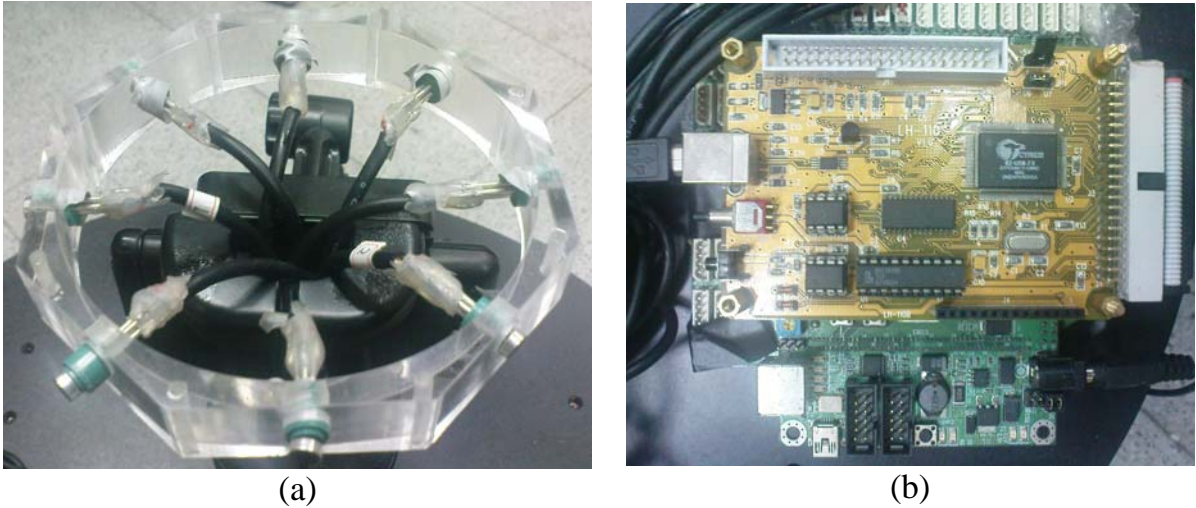


Figure 11. (a) The 8-Channel Digital Microphone (b) The FPGA and the USB Module

The experimental data was recorded in a 350cm*450cm room, with three static loud speakers playing human speech in English. The circular microphone array is of radius 5.5cm, so the far field assumption hold since the scale ratio between the room size and the array size is large enough. The robot is waking through a specifically design path to avoid some improper situations for DOA estimation and Bearings-Only SLAM. Figure 12 shows the environment that we record the sound data.



Figure 12. The Spatial Relation of the Speaker, the Robot, and the Laser Range Finder

4.2 Performance of offline calculated algorithm

The experiment was first done under an offline calculation to test the algorithm effectiveness. The program procedure for offline calculated FastSLAM is shown in Figure 13.

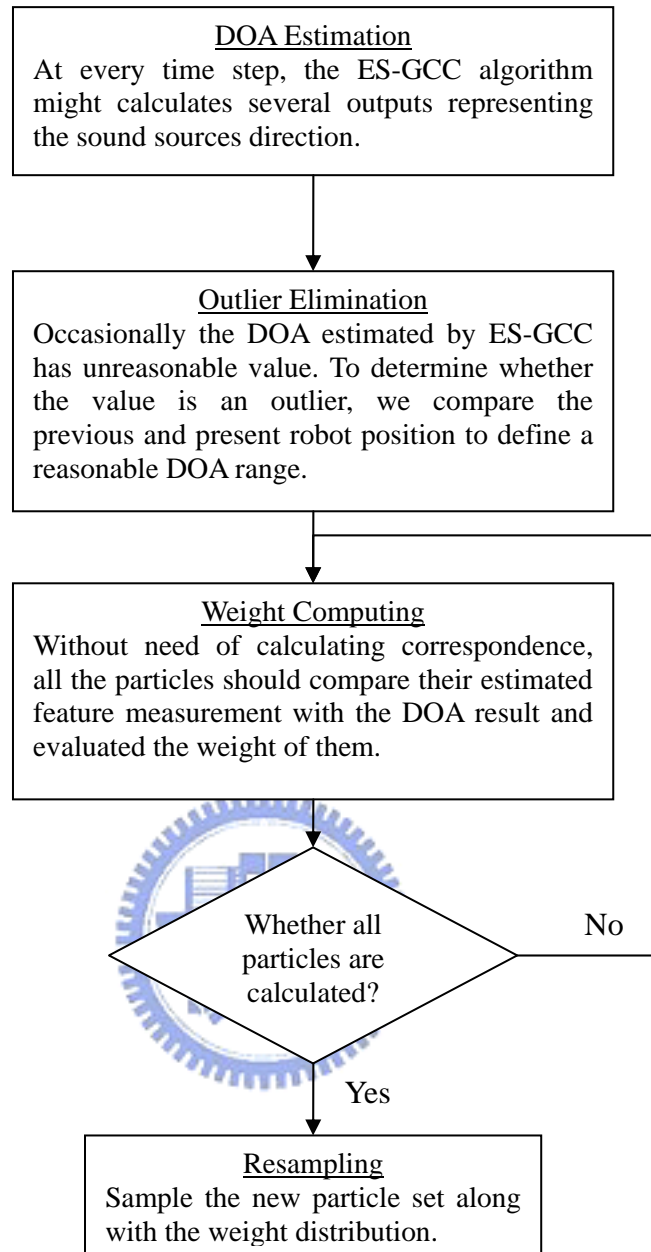


Figure 13. The Algorithm Procedure for Offline Calculation

As mentioned in section 2.3, the ES-GCC method estimates the DOA by using the least square method. The algorithm needs to accumulate a certain time frame to get solid delay information between microphones. Only those delay combination with reasonable sound speed generates DOA estimation. Furthermore, the DOA estimation is not necessary correct, which may be eliminated in the Outlier Elimination step. Upon all the reasons above, the DOA estimation won't be correct when the robot is

moving. So we program the algorithm to update the estimation only when the robot stops. The experimental results are shown below.

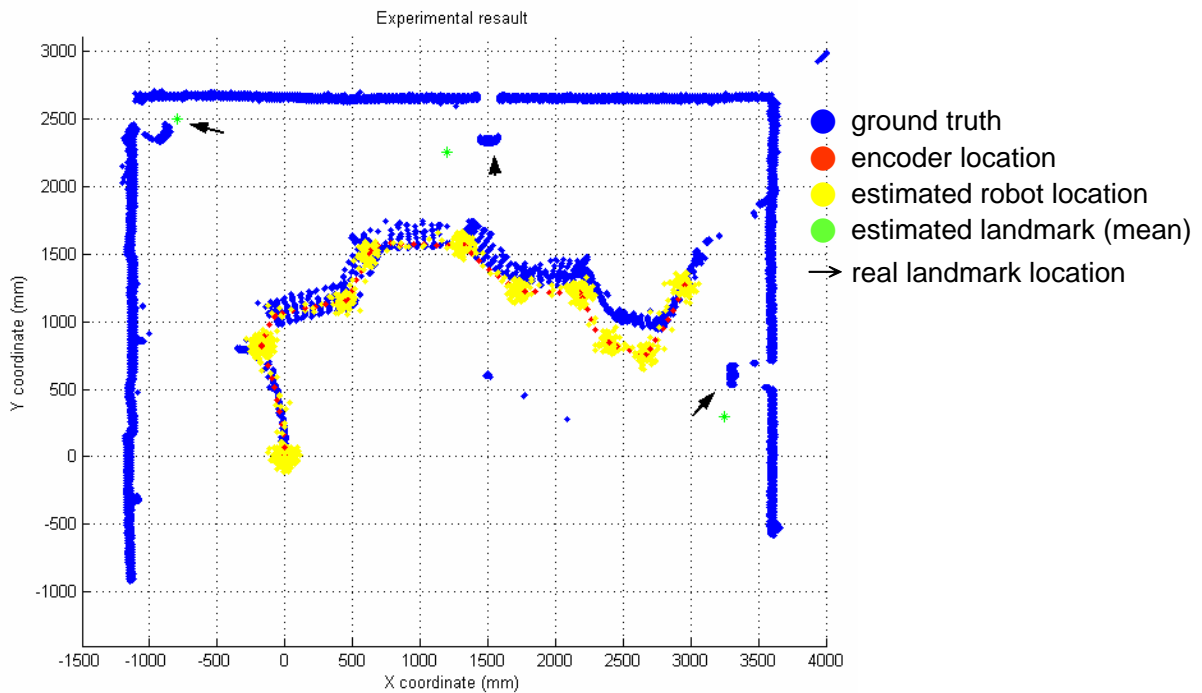


Figure 14. The Experimental Results of Offline Calculated FastSLAM

In Figure 14, the blue ground truth data are collected by a static laser range finder to compare the SLAM output and the real environment. The big yellow clusters are the waypoints that the robot stops to update the landmark estimation.

	Landmark 1	Landmark 2	Landmark 3	Average
Range Error	5 cm	22 cm	6cm	11cm
Range Error Rate	2.05%	7.90%	1.81%	3.92%
Bearing Error	2.81°	4.48°	5.03°	4.11°

Table 3. The Error Analysis of Offline Calculated FastSLAM

As in Table 3, the position estimation has larger error rate than the landmarks estimation. This phenomenon happens due to the joint uncertainty of the feature estimation and the encoder measurement. Conceptually, the position estimation is a sensor fusion result of the encoder and the DOA measurement. To adjust the believe ratio between DOA measurement and encoder, the covariance matrix of these two

sensor data should be modified.

4.3 Performance of online algorithm

The online algorithm here is not identical to the online SLAM mentioned in section 3.3. Here the “online” stands for the issue for real-time experiment. Some modifications of the FastSLAM algorithm are made for this application. The most significant adjustment is the removal of the resample state.

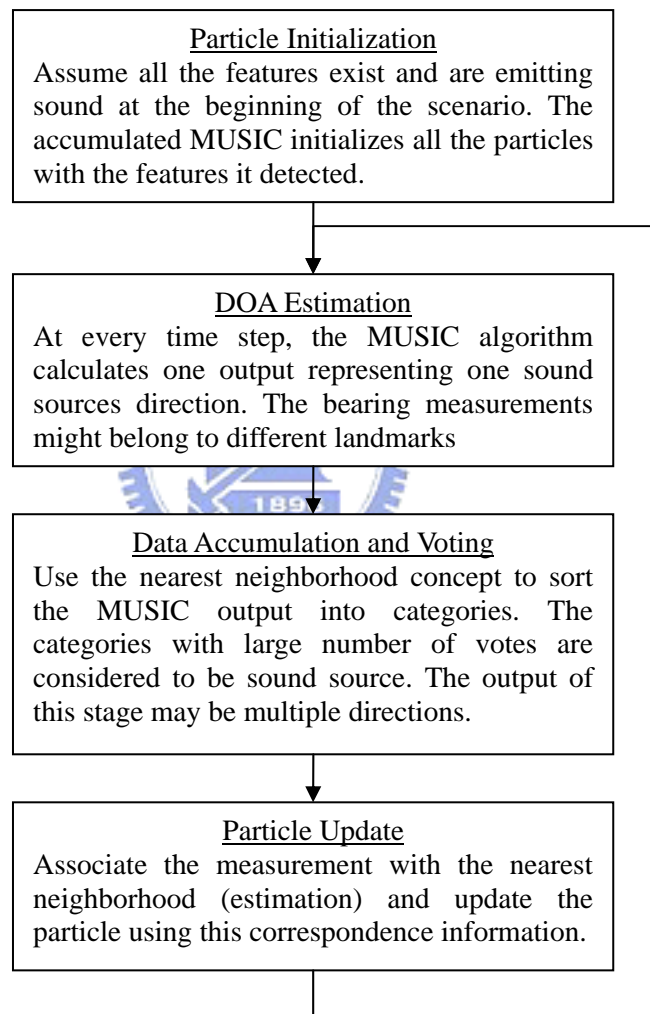


Figure 15. The Algorithm Procedure for Online Calculation

The pseudo procedure of the real-time experiment is illustrated in Figure 15. In order to release the calculation load of particle filter, the number of particles is decreased. Theoretically there is still a subset of the particle that is still having the

right correspondence. However, based on our experiment that this subset would not always exist, and hence the resample state will be forced to choose a wrong data association that is having relatively higher weight. This will fail the whole filter procedure.

Another adjustment is the DOA estimation method. We use the accumulated MUSIC to estimate the DOA instead of ES-GCC. MUSIC has a very good characteristic in finding single source. Under the situation of multiple speech sources with same scale of magnitude, MUSIC tends to find each of them sequentially. So, by accumulating the calculated directions, we sort this array using nearest neighborhood method. Only the clusters with sufficient amount DOA estimation pass through this step. The output of this step would be a robust DOA estimation.

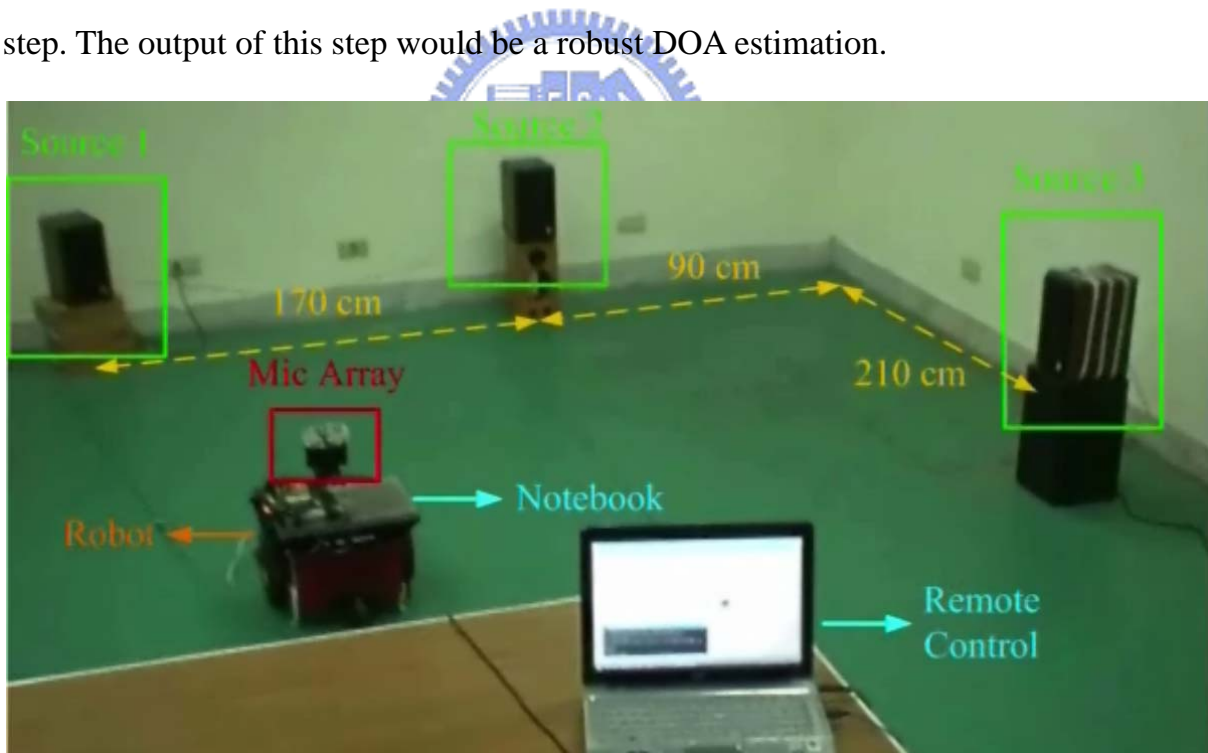


Figure 16. The Real-time Experimental Environment

Figure 16 shows how the speaker is distributed. The robot considers its start point as the origin and move through a path that doesn't have severe echo problem and

sound source overlap. We use 30 particles to describe the probability distribution.

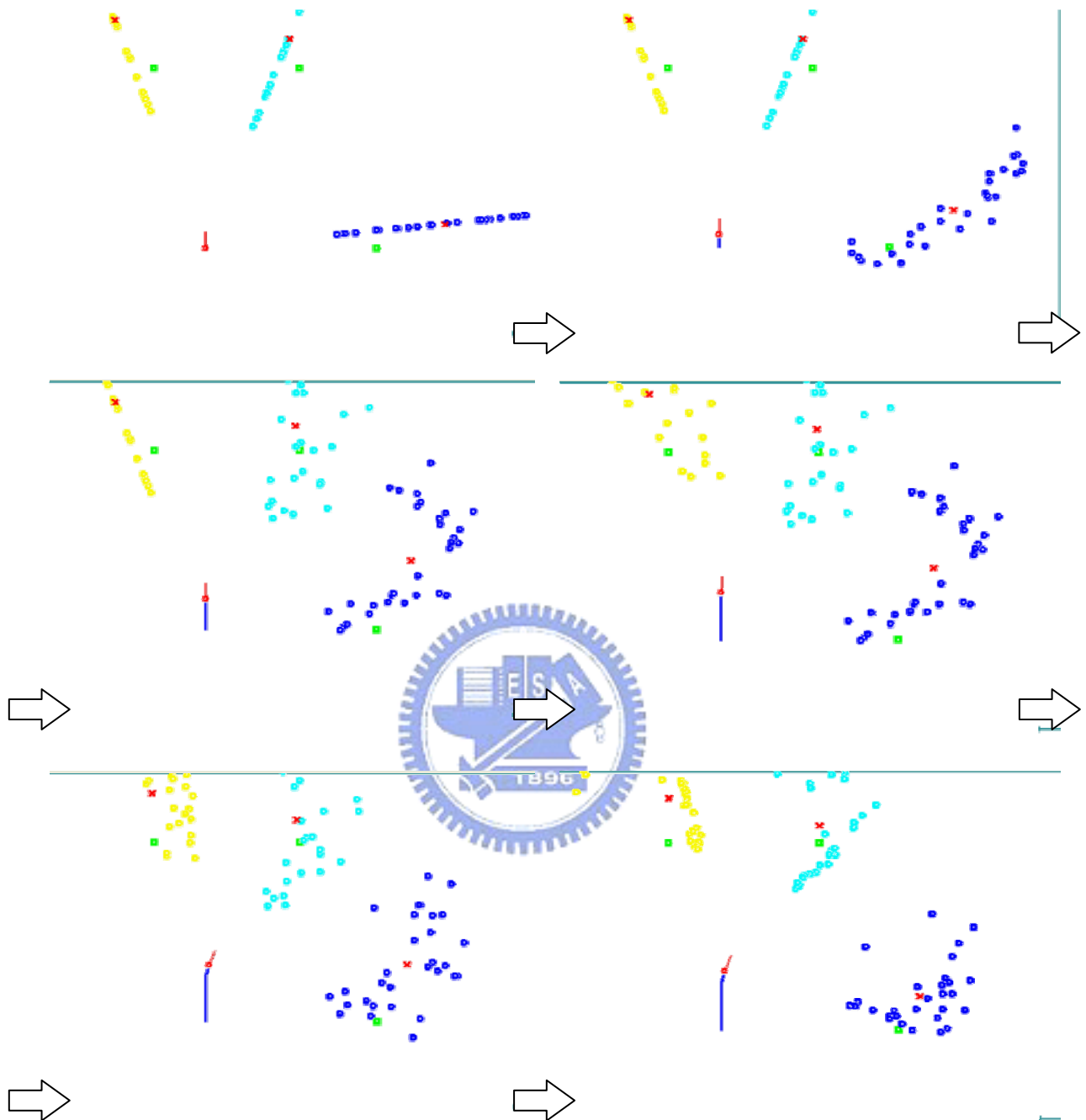


Figure 17. The Real-time Experimental Results

The red circle in Figure 17 is the robot position estimation. The yellow, light blue and blue circles are the distributed particles. The red crosses are the mean of each group of estimation, and the green square is the ground truth of the features.

The microphone array received the three angle estimations at the initialization

step, so three sets of particles were initialized. As the robot stopped at the next stop point, the accumulated MUSIC popped out one angle value. This angle was recognized to be a measurement of landmark 3. So the filter updated the estimation of landmark 3 for a while and headed to the next stop point. Afterward, the measurement of landmark 1 and 2 popped out. The filter was able to update arbitrary numbers of estimation group.

	Landmark 1	Landmark 2	Landmark 3	Average
Range Error	50 cm	2 cm	3 cm	18.33 cm
Range Error Rate	16.27%	0.70%	1.67%	6.21%
Bearing Error	1.74°	2.82°	4.76°	3.11°

Table 4. The Error Analysis of Real-time FastSLAM

As in Table 4, the average range error rate is 6.21%, and the average bearing error is 3.11°. The final result of the real-time FastSLAM is illustrated in Figure 18.

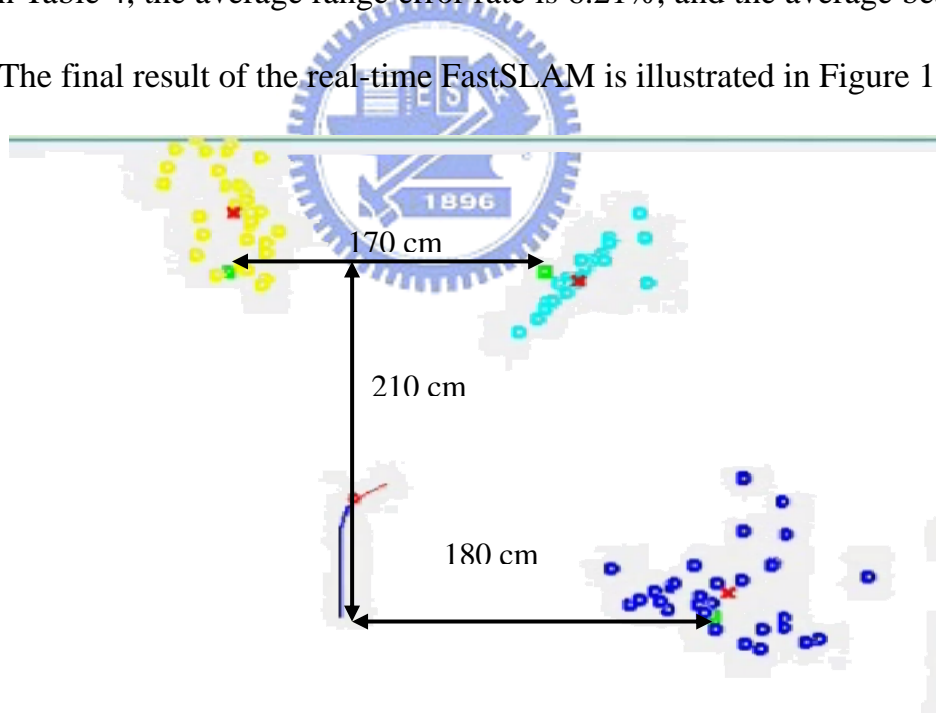


Figure 18. The Final Real-time Experimental Results

Chapter 5. Conclusion and Future Study

A combinational algorithm of DOA estimation and Bearings-Only SLAM is proposed in this thesis. The new algorithm is managed to deal with the landmark occlusion problem commonly faced in vSLAM. Using the ES-GCC or accumulative MUSIC allows us to estimate multiple DOAs within a short period of time frame. The theoretical knowledge of SLAM is presented in chapter 3 to explain the nondeterministic method that is used in the thesis. Figure 7 and figure 8 shows the simulation results of a particle filter.

The experimental result presented in chapter 4 show that the algorithm is applicable in offline calculation. Due to some hardware limitation, the real-time calculation procedure neglects the resampling part of the original algorithm and applies the nearest neighborhood concept to the data association. The simultaneous localization and mapping results are shown with range error of 6.21% and bearing error of 3.11° in average.

There are several areas for improvement. The more particles that are used in a particle filter, the more accurate it is. The real-time process could be modified in other method rather than decreasing the particle number. Also, the experiment in this thesis assumes that all the landmarks could be detected in the initial state. If there are new landmarks, the current algorithm is not able to localize them. An adjustable state vector should be used to intelligently modify the size of the vector to improve the SLAM robustness. Also, although the sound sources are detectable when they are NLOS, the DOA information does not hold the same characteristic. The specificity of NLOS sound sources should be studied.

REFERENCE

- [1] D. Johnson and D Dudgeon, “Array Signal Processing: Concepts and Techniques,” Prentice Hall, Englewood Cliffs, New Jersey, 1993.
- [2] Chia-Hsing Yang, “A Real-time Speech Purification and Voice Activity Detection System Using Microphone Array” National Chiao Tung University, Institute of Electrical and Control Engineering, Master Thesis, 2008
- [3]. C. H. Knapp and G. C. Carter, “The generalized correlation method for estimation of time delay,” IEEE Transaction on Acoustic Speech, Signal Processing, ASSP-24(4):320-327, Aug. 1976.
- [4]. R. O. Schmidt, “Multiple Emitter Location and Signal Parameter Estimation,” IEEE Transaction Antennas and Propagation, vol. AP-34, no. 3, pp.276-280, March 1986
- [5]. G. C. Carter, A. H. Nuttall, and P. G. Cable, “The smoothed coherence transform,” Proc. IEEE (Lett.), vol. 61, pp. 1497-1498, Oct. 1973.
- [6] S. Thrun, W. Burgard, D. Fox, Probabilistic Robotics, The MIT Press, Cambridge, 2005.
- [7] R. E. Kalman, “A New Approach to Linear Filtering and Prediction Problems,” Transactions of the ASME-Journal of Basic Engineering, 82 (Series D): 35-45, 1960
- [8] Cheng-Kang Wang, “Multiple Sound Source Direction Estimation and Sound Source Number Estimation,” National Chiao Tung University, Institute of Electrical and Control Engineering, Master Thesis, 2008
- [9] Source Website: <http://www.activrobots.com/ROBOTS/specs.html>