UNDERGRADUATE REPORT

Application of the Sound Localization Algorithm to the Global Localization of a Robot

by Kyung B. Ryu
Advisor: P.S. Krishnaprasad

UG 2001-5

Application of the Sound Localization Algorithm to the Global Localization of a Robot

Kyung B. Ryu
The Cooper Union for the Advancement of Science and Art
New York, NY 10003

Research Experience for Undergraduates (REU)
University of Maryland, College Park

July 30, 2001

Advisor:
Dr. P.S. Krishnaprasad
Intelligent Servosystems Laboratory (ISL)
Institute for Systems Research (ISR)
University of Maryland
College Park, MD 20742

Table of Contents

Tables:

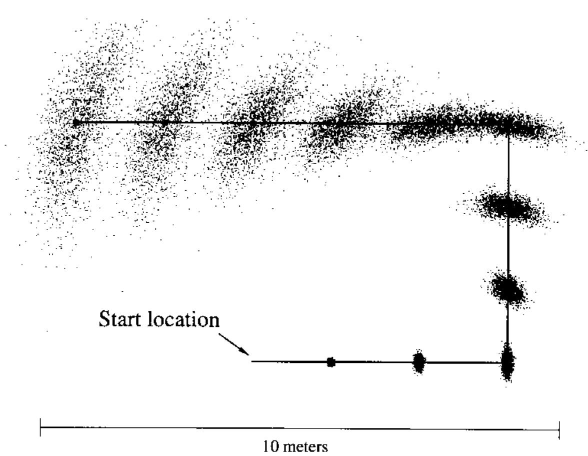
Figures:

Abstract

Sound localization is the process of determining the position of the sound source expressed by the azimuth and the elevation angles. Robot localization is the process of determining a robot's pose from sensor data. Based upon recent researches done on binaural sound localization algorithms in the Neural Systems Laboratory at the Institute for Systems Research, this research sought to apply localization of sound to automatically localize the robot in a small laboratory environment. By placing three sound sources at predetermined positions and letting the robot find the angles between the three sound sources by rotating in place to orient itself to each source in succession, the robot was able determine its current pose. The research successfully realized this model of robot localization. Simulation of the localization model was created to calculate range possible errors at different points in the map.

1 Introduction

Robot localization is the problem of finding out the pose (position and orientation) of the robot from sensor readings. The problem has been referred to as "the most fundamental problem to providing a mobile robot with autonomous capabilities" by some reference[1]. There are three categories of localization problems. The first and the most commonly pursued one is the *position tracking*. Here, the initial robot pose is known, and the problem is to compensate incremental errors in a robot's odometry. Solving this problem makes it necessary to input the robot's pose only once manually. The second is the *global localization problem*. This is more challenging than the first one. In this problem the robot is not told its initial pose, but instead has to determine it from scratch. This usually takes longer time than the first problem, because more data gathering and more analysis need to be done. This process can be used in conjunction with the position tracking to enable fully automated localization. The third problem is the *kidnapped robot problem*. This is the hardest problem of the three. In this problem, the robot is told its previous location but its previous location is totally off from its actual location. The robot has to perceive its prior knowledge is wrong and do a localization process on such a basis. This problem tests a robot's ability to recover from catastrophic localization failures. All these problems become extremely difficult if the robot is set in a dynamic environment [3].



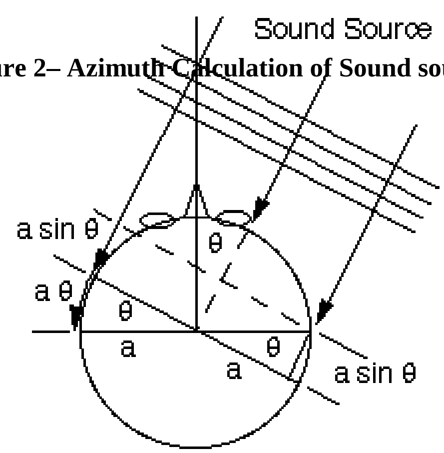
[Figure 1 – The Process of Position Tracking]

This research seeks to solve the global localization problem by implementing localization system such as the Global Positioning System (GPS) in lab environment. When the robot needs to localize itself globally, it will ask for localization algorithm and the localization algorithm will give the robot the pose in laboratory’s coordinates. However, this research localizes the robot through a method different from the GPS. Instead of localizing through determining the distance from three different known positions and triangulating the three distances to find out the location, this research seeks to localize through determining the angles between the known positions. In addition, instead of utilizing electromagnetic waves, the research will utilize sound waves. Using sound waves rather than electromagnetic waves may yield better results because sound waves are 10^5 orders slower than electromagnetic waves, and it is easier to measure a more precise time of travel.

2 Theoretical Background

There are many sound binaural localization algorithms available to localize sound. Two algorithms that were considered for this research were the Jeffress algorithm and the Stereausis algorithm. The Jeffress algorithm is a method of localizing sound solely by Interaural Time Difference (ITD). When the sound source is at the left of the listener, the left ear will receive the sound before the right ear does. The more the sound is to the left, the more delay there will be before the right ear will get the sound. This delay is called the ITD. The azimuth angle of the sound source can be determined by measuring the delay between the arrival of the sound signals to two ears. In contrast, the Stereausis algorithm is a biologically inspired algorithm and is more complex. It utilizes the cochlear filters to separate the sound source to different bandwidths and calculate ITD and Interaural Level Difference of each channel output and puts the results together with a neural network to yield one output. The Jeffress algorithm was chosen to localize sound in this research for its simplicity, fast computation, and easy implementation in C. Nullifying process (making the left and right microphone input the same by orienting the robot towards the source) was used in parallel with the Jeffress algorithm to yield better results.

Azimuth of the sound source can be determined by ITD:



[Figure 2– Azimuth Calculation of Sound source]

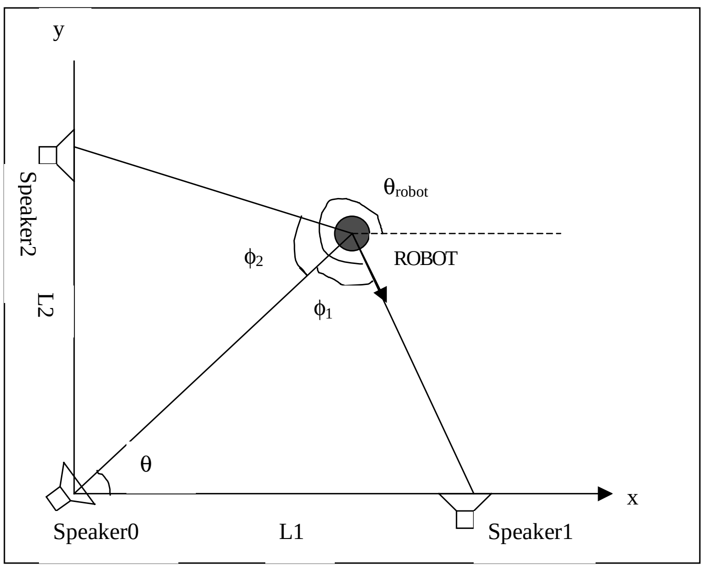
Let the speed of sound be **c** (about 343 m/s). Consider a sound wave from a distant source that strikes a spherical head of radius **a** from a direction specified by the azimuth angle θ to the right as shown in **Figure 2**. Clearly, the sound arrives at the right ear before the left, because it has to travel the extra distance $a\theta + a\sin\theta$ to reach the left ear. Dividing that by the speed of sound, we obtain the following simple (and surprisingly accurate) formula for the Interaural Time Difference:

$$ITD = \frac{a}{c}(\theta + \sin\theta) \quad , \ -90^\circ \le \theta \le +90^\circ$$

Thus, the ITD is zero when the source is directly ahead, and is a maximum of $(a/c)(\pi/2+1)$ when the source is off to one side. This represents a difference of arrival time of about 0.7 ms for a typical size human head, and is easily perceived. [4]

The next thing to be considered is the derivation of the formula which will yield the location of the robot given the azimuth of the sound. Since the robot used in this experiment can only move on the ground, only two-dimensional position (x and y) and orientation needs to be determined. It is proven that at least three speakers (sound sources) of known position are needed to figure out the pose the robot. This is also intuitive because one sound source will provide us with one-dimensional data (angle) and will solve for one independent variable, and there are three independent variables (x, y and θ_{robot}).

For simplification, the three speakers are positioned perpendicular to each other. Cartesian coordinate system is used because it is the most commonly used coordinate systems for mobile robots. Speaker0 is placed on the origin and speaker1 and speaker2 are respectively placed on the x-axis and the y-axis. Let the distance to speaker1 and to speaker2 from speaker0 be L1 and L2 respectively. Let ϕ_1 be the angle between speaker0 and speaker1, and let ϕ_2 be the angle between speaker0 and speaker2 (see the following figure).



[Figure 3 – Setup of the Three Speakers]

Then the formula that govern the relationship between the two angles and the pose of the robot will be given as the following (see APPENDIX I):

$$x = \frac{L1 + L1 \times \cot(\phi_1) \times \eta}{1+\eta^2}$$

$$y = \eta x$$

$$\theta_{robot} = \pi + \theta + \phi_1$$

where

$$\eta = \frac{L1 - L2 \times \cot\phi_2}{L2 - L1 \times \cot\phi_1} = \tan\theta$$

It is clearly shown that the pose of the robot (x, y and θ_{robot}) can be expressed in terms of L1, L2, and ϕ_1 and ϕ_2.

3 Application

The robot used in this research was the Super Scout II by Nomadic Technologies. It is equipped with a Pentium 233 MHz processor and runs on Linux platform. The sound sampling was done at 44191Hz stereo through a Sound Blaster Pro™ soundcard for all experiments. The sound was picked up by two microphones and was primarily filtered through analogue amplifier/filters that were made by Sean Andersson before being digitally sampled.

3.1 Sound Localization

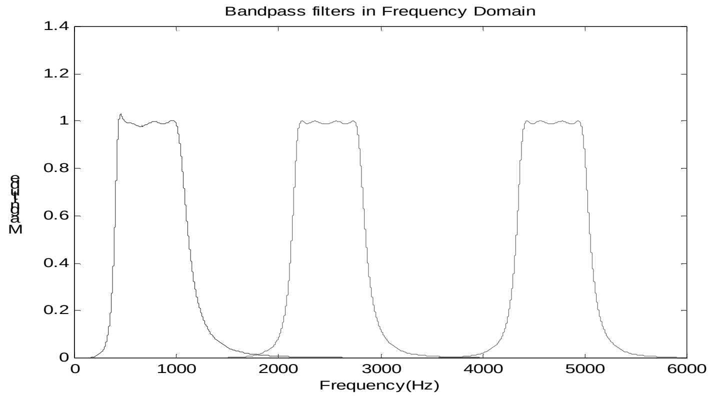
3.1.1 Generation of the Sound

Before sound can be localized, there need to be sound that can be played back from the speakers. It is not known which sound works best for the Jeffress algorithm given the microphones, acoustic characteristics of the room and other variables specific to the laboratory settings so two sets of three different sounds were designed to be tested to find out the best localizing sound. One set was created by feeding a random white noise through band pass filters of three different bands (see Table 1). The other set was generated by cosine functions. Noise was generated for comparison because previous sound research has shown that white noise is best localized [2]. Tones were produced as well because it was thought that correction can be made to the retrieved sound data with noise due to its simplicity. The sound waves were generated in .wav format in MATLAB with a custom-made genwav() function (see APPENDIX III). Shown below are the passband of the bandpass filters to design the noise set and the frequency of the tones:

Noise			Tones	
	Pass-band	filename	frequency	Filename
Bandpass Filter1	441~992.25Hz	Low.wav	200Hz	Cos1.wav
Bandpass Filter1	2205~2756.25Hz	Med.wav	400Hz	Cos2.wav
Bandpass Filter1	4410~4961.25Hz	High.wav	600Hz	Cos3.wav

[Table 1 – The pass band of the bandpass filters and the frequency of the tones]

The following is the frequency response of the three band pass filters that were used to generate three different sounds.



[Figure 4 – Frequency Response of Three Bandpass Filters]

The frequencies of the tones (cosine waves) were carefully picked to avoid any ambiguity when calculating ITD. In other words, maximum ITD will not exceed half the period of the highest frequency tone. As it can be seen in Figure 4, the bandpass filters were carefully designed so that there would be little overlapping pass bands between any two filters while keeping the amplitude of the pass band relatively flat.

3.1.2 Localization of sound using ITD

The highest sampling rate that was possible for stereo sampling for the given sound card (44191Hz) was used to ensure the highest possible precision. Since one sampling period is 1/44191 sec (22.6 μsec) and the maximum time delay possible for one sound source is:

$$\frac{Diameter}{speed of\, sound} = \frac{0.308m}{340m/s} \simeq 0.906ms$$

This time is corresponds to roughly 40 sampling periods. So there would be a maximum of 40 sample delays between the two sets of microphone inputs in case the sound is 90° left or right. With the coarse estimate that time delayed linearly increases as the azimuth angle increases, there would be 41 detectable sample delays over 90° evenly. This estimate shows that 2.25° is the minimal azimuth angle change possibly detectable. It turns out that the time delay does not increase linearly and that this method can detect lesser than 2.25° difference at low azimuth angles while it can't detect more than the angle at high azimuth angles (see Table 2). In fact at low azimuth angles it can detect up to 1.43° of azimuth angle change while at high azimuth angles nearly 11° in change can be unnoticed.

Index	Azimuth	Index	Azimuth	Index	Azimuth
0	0	14.0000	20.4720	28.0000	44.3869
1.0000	1.4315	15.0000	22.0078	29.0000	46.4258
2.0000	2.8639	16.0000	23.5603	30.0000	48.5440
3.0000	4.2981	17.0000	25.1314	31.0000	50.7549
4.0000	5.7351	18.0000	26.7231	32.0000	53.0756
5.0000	7.1756	19.0000	28.3373	33.0000	55.5288
6.0000	8.6207	20.0000	29.9764	34.0000	58.1457
7.0000	10.0714	21.0000	31.6430	35.0000	60.9711
8.0000	11.5286	22.0000	33.3401	36.0000	64.0737
9.0000	12.9934	23.0000	35.0709	37.0000	67.5689
10.0000	14.4669	24.0000	36.8392	38.0000	71.6810

11.0000	15.9503	25.0000	38.6494	39.0000	76.9830
12.0000	17.4447	26.0000	40.5066	40.0000	87.8341
13.0000	18.9515	27.0000	42.4167	41.0000	90.0000+

[Table 2 – The azimuth angle corresponding to the index of delays]

The ITD is measured in multiples of sampling period. Since the ITD is expressed in integers only, we will call the integer number which pertains to ITD, *index* for convenience. So the index is measured through cross correlation after the left and right samples are collected. The correlation coefficient for the left sample and the right sample delayed by j number of sampling period is calculated at different j (-maxshift ≤ j ≤ maxshift). Since the left microphone and the right microphone input are same sound source, the correlation coefficient will be maximum when the phase difference between the two samples is zero. Therefore the j with maximum correlation coefficient is the time that left signal lags or leads ahead of the right signal. If j is negative, that means that left signal lags behind the right signal by that many sampling periods, and if j is positive, it means that the left signal leads the right signal by that number of sampling periods. The correlation coefficient for the two signals is calculated as the following:

$$c_j = \sum_{i=1}^{n} x_i y_{i+j} \quad (\text{-maxshift} \le j \le \text{maxshift})$$

where x is the left signal and y is the right signal.

3.1.3 Localization Results

The localization process was tested with the robot and six different sounds. The measurements of the index were made at azimuth angles, 0 degree, 4 degrees to the left/right and 45 degrees to the left/right. The localizations at these angles were performed at two different distances: at 2.5ft and at 7ft. A hundred samples were taken at each position but sound that wasn’t strong enough were cut off from printing results by the program so at some positions you’ll see less than a hundred data. Only ten results from each location are printed here for succinctness.

Noise1			Noise2			Noise3		
0	LEFT 4°	LEFT 45°	0	LEFT 4°	LEFT 45°	0	LEFT 4°	LEFT 45°
0	2	27	1	2	27	-1	2	20
0	2	27	1	2	27	-1	2	20
0	3	27	1	2	27	-1	2	20
0	2	27	1	2	27	-1	2	20
0	2	27	1	2	27	-1	2	20
0	2	27	-18	2	27	-1	2	20
0	1	27	1	2	27	-1	2	20
0	1	27	1	2	27	-1	2	20
0	1	27	1	2	27	-1	2	20
0	2	27	-18	2	27	-1	2	20
Wave1			Wave2			Wave3		
0	LEFT 4°	LEFT 45°	0	LEFT 4°	LEFT 45°	0	LEFT 4°	LEFT 45°
-8	2	14	-27	2	27	4	2	27
9	0	26	-27	-14	27	3	6	27
-6	-20	27	-27	-24	-27	4	5	27

-4	6	15	-27	-26	-27	4	3	25
7	4	27	-27	-9	-27	4	5	22
-13	-6	23	-27	-8	-27	4	5	22
7	12	18	-27	-27	-27	4	5	22
0	-7	27	-27	-25	-27	4	5	21
-11	-3	14	-27	-21	-27	4	5	20
9	8	21	-27	-21	-27	3	6	20

[Table 3 – Localization of sound at 2.5ft]

Noise1			Noise2			Noise3		
0	RIGHT 4°	RIGHT 45°	0	RIGHT 4°	RIGHT 45°	0	RIGHT 4°	RIGHT 45°
0	-2	-25	1	-2	-26	-19	-2	-20
-1	-3	-26	-1	-4	-26	-24	-3	-20
0	-2	-25	-1	-4	-26	22	24	17
0	-2	-25	-1	-4	-26	-24	-3	-20
-22	-2	-24	-1	-4	-26	22	24	17
-1	-3	-25	0	-4	-26	-24	-3	17
22	-2	-26	0	-23	-26	-24	-3	-20
-22	-2	-26	0	-4	-26	22	-3	-20
0	-2	-25	0	-4	-26	-24	-3	-20
0	-2	-25	0	-4	-26	22	-3	-20
Wave1			Wave2			Wave3		
0	RIGHT 4°	RIGHT 45°	0	RIGHT 4°	RIGHT 45°	0	RIGHT 4°	RIGHT 45°
1	1	-27	0	-6	-27	1	1	-23
-20	-1	-8	-7	-3	-25	3	1	-18
3	-27	-19	-9	-10	-27	4	3	-11
-1	6	-27	-7	-6	-27	4	1	-10
-20	-13	-13	-13	-21	-27	3	4	-13
1	-23	-27	-17	-19	-27	4	3	
-11	0	-27	-18	-27	-27	3	4	
-19	-17	-11	-18	-24	-27	5	5	
2	-12	-27	-19	-21	-27	6	5	
-15	-4	-19	-20	-27	-27	5	5	

[Table 4 – Localization of sound at 7ft]

A lot can be said about the result but in short, the best sound for localization is the noise1 (low.wav) or noise2 (med.wav). Noise1 performed best at close range being the most consistent and correct at all three angles. But it seems to falter a little bit at medium range showing several off measurements at zero azimuth angle at 7ft. The tones seem to perform terribly both being incorrect and oscillating back and forth between measurements each time. It is noticeable that at seven feet away, the wave3 was only able to output five samples due to low sound level. Also, note that when the sound is coming from the left, the indices are positive denoting that left signal leads the right signal and vice versa, they are negative when the sound source is coming from the right.

3.2 Nullifying sound

The term “nullify” is used loosely in this paper to mean making left and right microphone input equal. Nullifying sound is the process of finding the azimuth angle from current orientation by repetitively sampling and rotating in place to orient the robot toward the sound until the robot has succeeded in making left and right sound equal (no phase shift, index = 0). This process was used to get accurate measurement of the azimuth angle of the sound source. As it can be seen from Table 2 the Jeffress model of detecting ITD for azimuth angle is very inaccurate at high azimuth angles not being able to detect differences up to 10 degrees (see corresponding azimuth angles at index of 39 and 40). Instead of getting the azimuth angle from one measurement, if the angle is calculated from nullifying sound precision of 1.43 degree becomes possible. The only problem with this approach is that some error is introduced when turning to the sound, since the measurement has to be done. The magnitude of this error is relevant, but the specific number is not known due to lack of experiments. Other benefits of this approach include: greater accuracy and precision by nullifying bias, being able to distinguish whether the sound is coming from behind for front (the normal ITD algorithm cannot distinguish this). These advantages outweigh the errors introduced by the measurement of rotation angles.

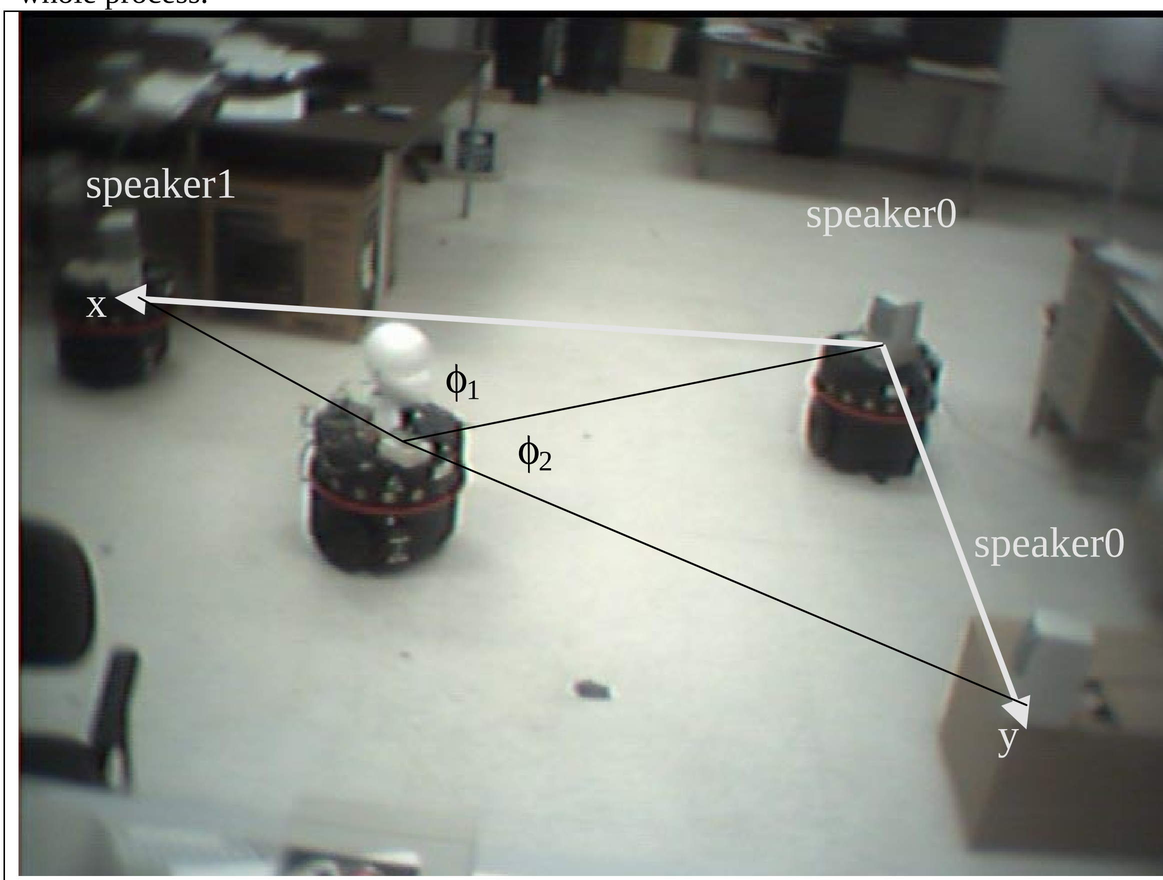
3.3 Robot Localization

Now that the problem of localizing the sound is solved, the only thing that remains is localization of the robot by localizing the sound from three different locations. The three speakers are set up as shown in Figure 3. The implementation of it in the real world is shown in Figure 5A. The x and y axis are drawn in the picture to help seeing the implementation. Each speaker is hooked up to a robot because process of localization of sound can only localize to one sound at a time and sound output must be controlled. The robot that is hooked up to a speaker acts a server that plays and stops the sound as requested by the client robot (the robot that needs to localize). This system was also chosen because it can be developed into many other interesting models: the groups of robots can be moving around, and all the robots can be localized if only three robots know their current position.

The localization process was done in the following sequence:

1. Nullify to speaker2
2. Nullify to speaker0 and record the angle that it turned to the left (ϕ_2)
3. Nullify to speaker1 and record the angle that it turned to the left (ϕ_1)
4. Calculate x, y, and θ_{robot} according to ϕ_1 and ϕ_2

Doing localization in this fashion saves unnecessary rotations and reduces errors. For example if speaker0 was nullified first, it needs to nullify to either speakers and make an extra turn to speaker0 to measure the other angle. Figure 6 shows the process of the whole process.



[Figure 5A – Nullifying to Speaker 2]

[Figure 5B – Nullifying to Speaker 0]

[Figure 5C – Nullifying to Speaker0 success]

[Figure 5D – Nullifying to Speaker1]

[Figure 5E – Nullifying to Speaker1 success]

4 Simulation

4.1 Designing the Simulation Process

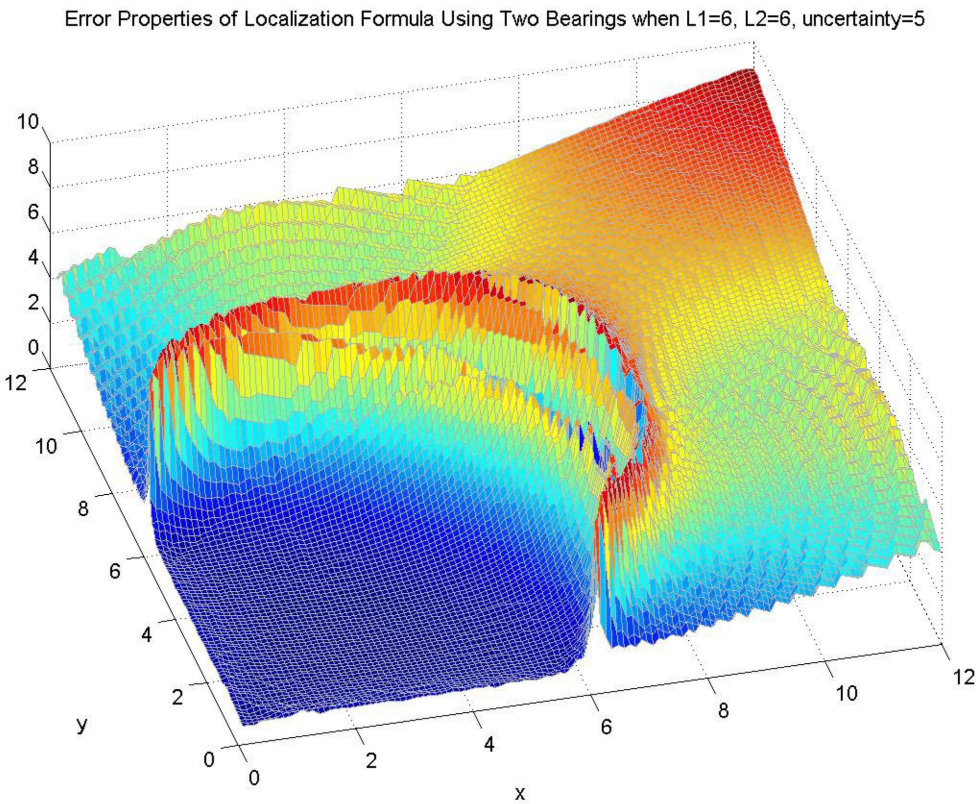
The formula to derive the pose of the robot is not linear to the two measurements ϕ_1 and ϕ_2. Therefore it is hard to predict what the errors will be at certain locations. To see what it would be like through actual experimenting will take too much time and effort. So the best solution to this problem was to design a simulation program that would estimate the errors at various locations in the test field. To do this, I've designed a MATLAB function that would plot the maximum possible error at every point of the map given L1, L2 and the uncertainty. The algorithm of the simulator is as the following:

1. Calculate the angle ϕ_1 and ϕ_2, if it were at position x, y
2. Add noise (uncertainty) and recalculate positions x, y corresponding to the new sets of ϕ_1 and ϕ_2 angles.
3. Find the maximum distance error between the original position and the newly calculated positions.
4. Iterate through steps 1-3 until it has calculated errors for all the positions

I've designed another simulation program plots the average error of the whole field at different placements of the two speakers. This was made based on the first simulation program. The purpose of the second simulation program was to find out the optimum placement of the two speakers.

4.2 Results of Simulation

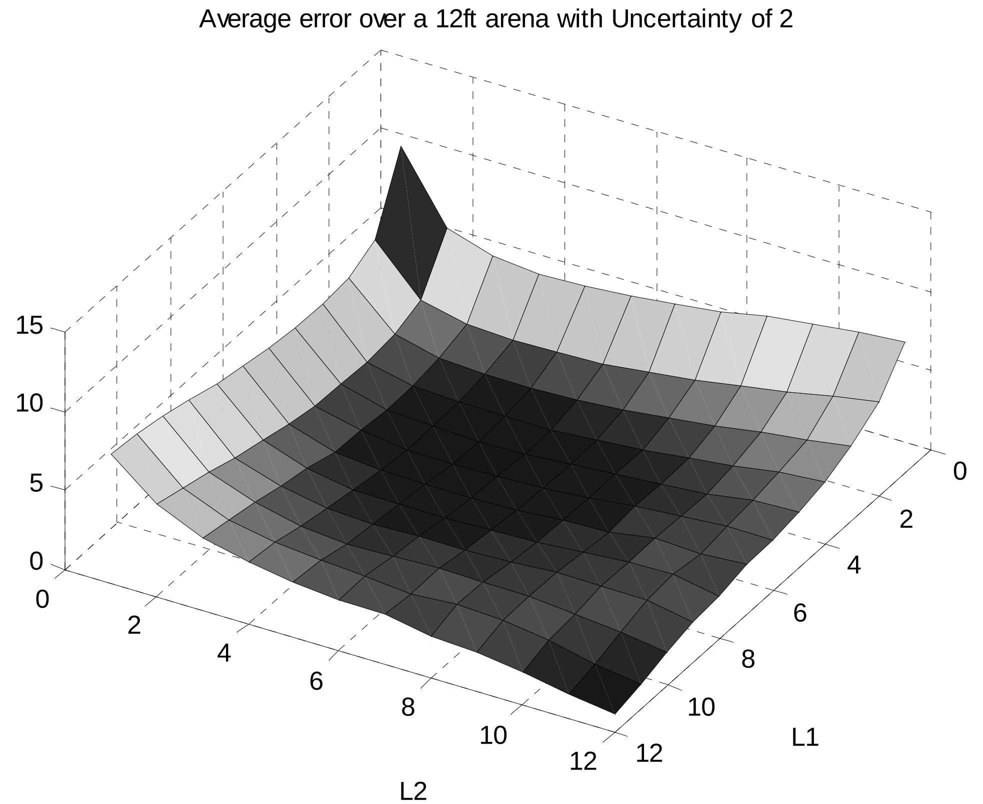
This is what the resulting graph from the first simulation program looks like:



[Figure 6 – Error Estimation from Simulation]

Intuitively, as the robot is placed further away from the origin, the error becomes larger. However, it is interesting to see such a large error when the robot is very close to the speaker around 7 ft range that forms a large arc around the inner area. The ϕ_1 and ϕ_2 angles at these locations are all similar varying about 2° at most. Therefore small error in ϕ_1 and ϕ_2 would lead to a result in pose that is totally off from the actual pose of the robot.

The following graph is derived from the second simulation program:



[Figure 7 – The Error Plot at Various Combinations of L1 and L2]

This was somewhat expected, but it is reassuring that the optimum placement of the speakers are optimum when they are farthest possible distance. The reason that error is so low for the case when L1=12ft, L2=12ft is from the fact that the arc of high errors are not included in the test field of 12x12 area.

5 Conclusion and Future Works

The model that was developed by Dr. Krishnaprasad—the model of localizing the robot from three sound sources, was realized and implemented to the robots. However, because of lack of precision of the sound localization process and relatively uncertain error properties that the formula exhibits (the arc of huge errors around the speakers), it seems improbable to implement the model to a laboratory use directly. The average error over the whole region in the simulation with the optimum speaker positioning and reasonable uncertainty (±5°) turned out to be 2.2ft, and this is too high of an error to make this method possible to use as localization process in an environment of 12x12ft area.

Many developments can be made to the model, however. Just using a better soundcard that can sample sound at higher frequency will yield better sound localization and yield more accurate pose of the robot. Improvements can also be made by using better sound localization algorithms such as stereausis algorithm, so the robot can measure more accurate and precise ϕ_1 and ϕ_2. In addition, towards the end of the experiment, it was found that pseudorandom sound signals of sound can also be used to estimate the distance from the sound source to the microphone. This information, if retrieved, can be also used to correct some errors made by the model studied in this paper.

Acknowledgments

I thank Dr. Krishnaprasad for his personal guidance and advice. I extend my thanks to Sean Andersson, Clifford Knoll, and Fumin Zhang for their countless many accounts of help and encouragement. Also great thanks to Diane Ihasz who organized this whole program and helped me many times in settling down this new environment.

Reference:

[1] I.J. Cox. Blanche—an experiment in guidance and navigation of an autonomous robot vehicle. *IEEE Transactions on Robotics and Automation*, 7 (2):193-204,1991

[2] E. Grassi and S.A. Shamma. A biologically Inspired, Learning, Sound Localization Algorithm. In *Proceedings of the Conference on Information Sciences and Systems*, The Johns Hopkins University, March 21-23,2001

[3] S. Thrun, etc. Robust Monte Carlo Localization for Mobile Robots. To appear on *Artificial Intelligence*, Summer 2001.

[4] Unknown. Azimuth Cues. *http://phosphor.cipic.ucdavis.edu/CIL_tutorial/3D_psych/azimuth.htm*

Bibliography

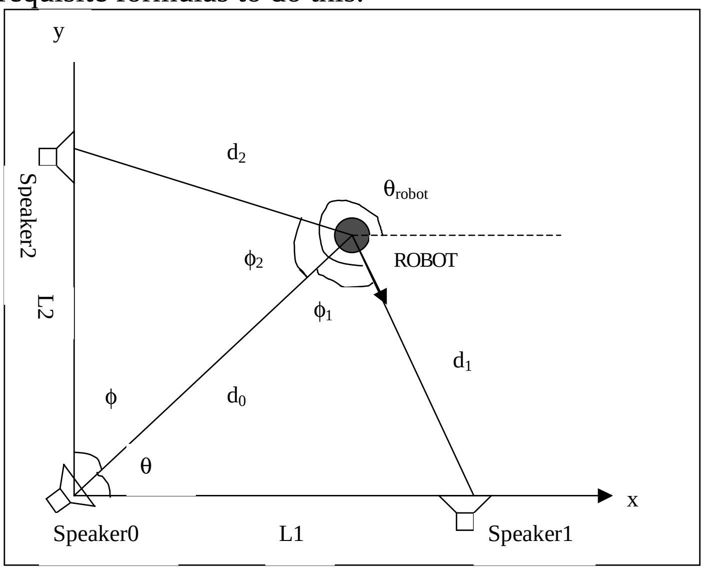
Alberto Leon-Garcia. *Probability and Random Processes for Electrical Engineering*. Second Edition. Pages 232-241. Addison-Wesley Publishing Company, New York, 1990

J.N. Little L. Share. *Signal Processing Tool Box*. The Math Works Inc. 1993

Appendix I: Derivation of the Formula

Mathematical proof done by Dr. P.S. Krishnaprasad

Consider a laboratory environment equipped with 3 sound sources distinguishable by spectral signatures. A binaural robot equipped with a sound localization algorithm can localize itself within the laboratory only using source bearings. We derive the localization formula by appealing to basic properties of triangles in the plane. It is arrived that the robot can rotate in place to orient itself to each of the three sources in succession. The robot can determine two angles ϕ_1 and ϕ_2 between the three directions to the three directions to the three sources. These two angles together with the fixed baselines between the sound sources is sufficient to determine the (x,y) coordinate of the robot. We derive below the requisite formulas to do this.



The robot is at point P(x,y). The baselines between the sources located at the points 0, 1 and 2 are L1 and L2. The origin of the coordinate system is at 0 and 01 is the x-axis and 02 is the y-axis. The axes are assumed to be perpendicular.

It is clear that if the distances d0 and angle θ are determined then the x, y coordinates can be determined without ambiguity. Consider the triangles 02P and 01P. Apply the sine rule to each of these two triangles to obtain:

$$\frac{L2}{\sin\phi_2} = \frac{d_2}{\sin\phi} = \frac{d_0}{\sin(\pi-\phi-\phi_2)} = \frac{d_0}{\sin(\phi+\phi_2)} \ldots(1)$$

$$\frac{L1}{\sin\phi_1} = \frac{d_1}{\sin\theta} = \frac{d_0}{\sin(\pi-\theta-\phi_2)} = \frac{d_0}{\sin(\theta+\phi_2)} \ldots(2)$$

From (1) and (2),

$$\frac{\sin(\phi+\phi_2)}{\sin\phi_2}L2 = \frac{\sin(\theta+\phi_1)}{\sin\phi_1}L1 = d_0 \ldots(3)$$

Hence,

$$\frac{\sin\phi\cos\phi_2+\cos\phi\sin\phi_2}{\sin\phi_2}L2 = \frac{\sin\theta\cos\phi_1+\cos\theta\sin\phi_1}{\sin\phi_1}L1 = d_0 \ldots(4)$$

But $\theta + \phi = \pi/2$

$\Rightarrow \sin(\phi) = \cos(\theta)$

$\cos(\phi) = \sin(\theta)$

Substituting in (4),

$$\frac{\cos\theta\cos\phi_2+\sin\theta\sin\phi_2}{\sin\phi_2}L2=\frac{\sin\theta\cos\phi_1+\cos\theta\sin\phi_1}{\sin\phi_1}L1=d_0$$

simplifying,

$$\Rightarrow \sin\theta[\cot\phi_1\times L1-L2]=\cos\theta[\cot\phi_2\times L2-L1] \quad \ldots(5)$$

$$\Rightarrow \tan\theta=\frac{\cot\phi_2\times L2-L1}{\cot\phi_1\times L1-L2}$$

$$\theta=\arctan\left(\frac{\cot\phi_2\times L2-L1}{\cot\phi_1\times L1-L2}\right)\ldots(6)$$

The actual coordinates of the robot are determined from the formulas,

$$x=d_0\cos\theta$$
$$y=d_0\sin\theta=x\tan\theta \quad \ldots(7)$$

Let tan θ be η.

Then substituting d0 into (7) we get,

$$x=\frac{L1+L1\times\cot(\phi_1)\times\eta}{1+\eta^2}$$

$$y=\eta x$$

Appendix II: Correlation algorithm

The correlation coefficient is the covariance of two data sets divided by the product of their standard deviations:

$$\rho_{X,Y}=\frac{\mathrm{cov}(X,Y)}{\sigma_x\times\sigma_y}$$

where

$$\sigma_x{}^2=\frac{1}{n}\sum(X_i-\mu_x)^2$$

$$\mathrm{cov}(X,Y)=\frac{1}{n}\sum(X_i-\mu_x)(Y_i-\mu_y)$$

You can use the correlation coefficient to determine whether two ranges of data move together — that is, whether large values of one set are associated with large values of the other (positive correlation), whether small values of one set are associated with large values of the other (negative correlation), or whether values in both sets are unrelated (correlation near zero).