Lab1: Data Manipulation and Modeling

Josephine King, Peter Johnson

***Abstract*—This report details the characterization and analysis of lidar and GPS data. In MATLAB, the lidar data is used to construct a sensor model and the the GPS measurements are transformed to a coordinate system and their variances are explored. Finally, the sensor model is used to implement Baye's rule for different states based on a single range measurement.**

I. Introduction

FOR Lab 1, we were provided with data from a Velodyne Puck VLP-16 3D laser scanner and a SparkFun NEO-M8P-2 RTK GPS. The data were obtained by setting the lidar in a stationary position in the Parsons building courtyard. The roll and pitch angles are approximately zero. The yaw angle is 90 degrees corresponding to facing north. In the data sets, the only measurements logged are those with elevation angle zero, and one of three azimuth angles [-90, 0, +90].

II. Lidar Data

First we constructed histogram plots which showed the frequency of measurements taken for each azimuth angle (See Figues 1, 2, 3.) We tried multiple bin sizes for each data set in order to get the data to appear as normally distributed as possible.

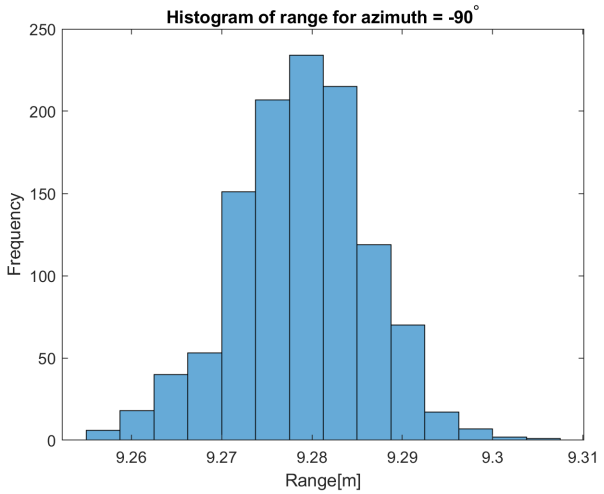

Fig. 1. Range data for azimuth = $-90°$. Bin width of 3.75 mm

III. Sensor Model

We based our model for the azimuth = $-90°$ data on the beam model for range finders presented in Probabilistic Robotics. We chose to model our data using a single Gaussian (normal distribution). This represents an accurate measurement with local noise. Following the notation in the book, z_t^{k*} denotes the "true" range of the object, whose measurement is

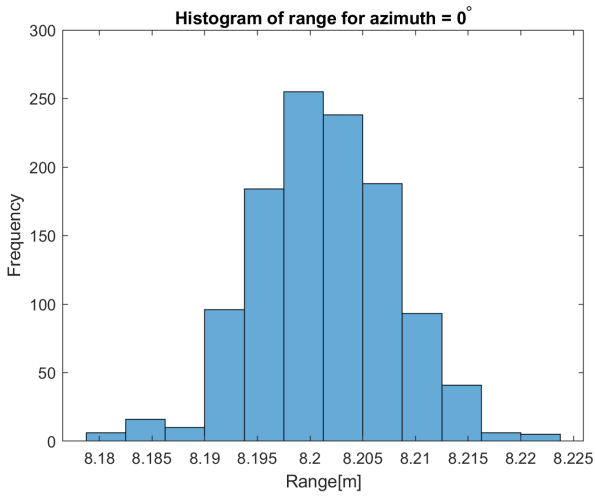

Fig. 2. Range data for azimuth = $0°$. Bin width of 3.75 mm

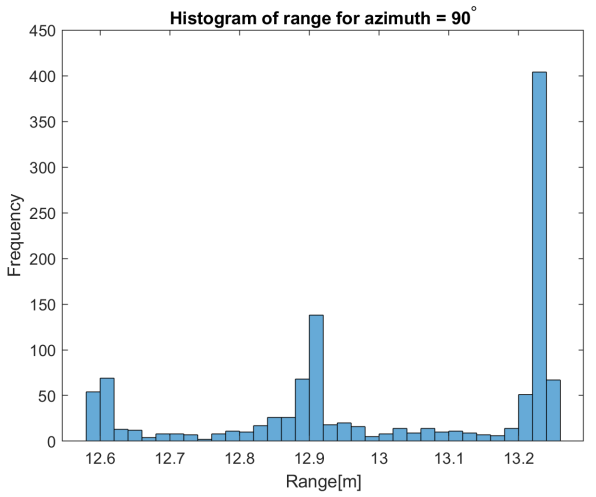

Fig. 3. Range data for azimuth = $90°$. Bin width of 20 mm

z_t^k. The value that the sensor measures is subject to error which arises from the limited resolution of the sensor, atmospheric noise and so on. We model this noise using a Gaussian with mean $\mu = z_t^{k*}$ and standard deviation σ_{hit}. We refer to this Gaussian as p_{hit}. We used a MATLAB function to find the parameters μ and σ_{hit}. We can compare this fit to the data in Figure 4

Since our fit appears to be good, we can then then derive a function $p(z|x)$ based on this fit. We follow Thrun's development of the function to get

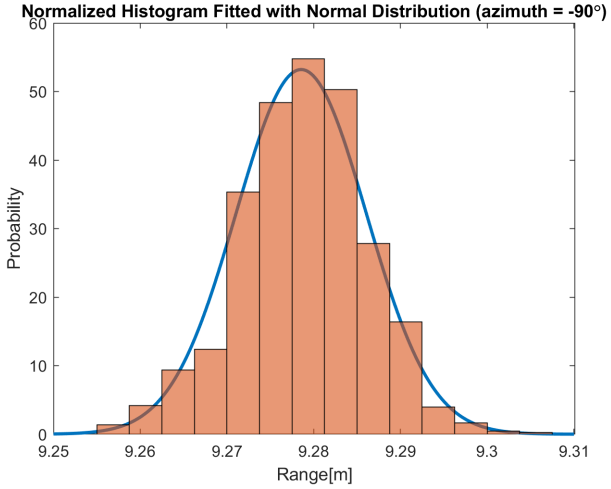

Fig. 4. Gaussian distribution fitted to azimuth = $-90°$ data. $\mu = 9.2786$ $\sigma_{hit} = 0.0075$

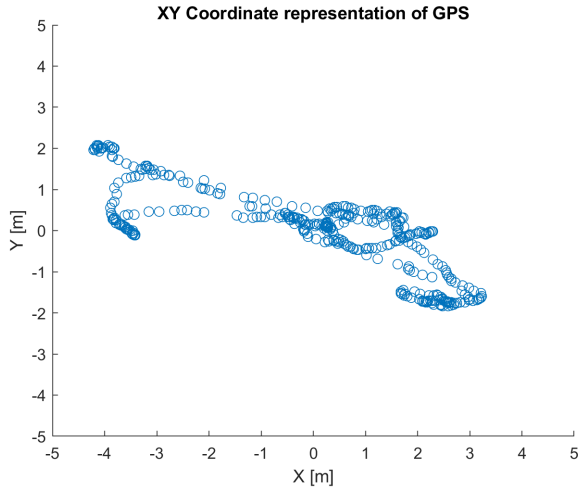

Fig. 5. Location data for azimuth = 0°. Origin is at 34.1064, -117.7120

$$p_{hit}(z_t^k|x_t, m) = \begin{cases} \eta\mathcal{N}(z_t^k; z_t^{k*}, \sigma_{hit}^2) \text{ if } 0 \leq z_t^k \leq z_{max} \\ 0 \text{ otherwise} \end{cases}$$

where z_t^{k*} is calculated from x_t (the state) and m (the map) using beam casting. In our implementation, we determine the appropriate z_t^{k*} based on x_t beforehand, and simply pass the value to the function.

$$\mathcal{N}(z_t^k; z_t^{k*}{}_{hit}) = \frac{1}{\sqrt{2\pi\sigma_{hit}^2}} e^{\frac{-1}{2}\frac{(z_t^k - z_t^{k*})^2}{\sigma_{hit}^2}}$$

This term is just the Probability Density Function (PDF) of a normal distribution. A PDF provides *relative likelihood* that the value of the random variable would equal that sample when compared to another value.

$$\eta = \Big(\int_0^{z_{max}} \mathcal{N}(z_t^k; z_t^{k*} dz_t^k\Big)^{-1}$$

This is the Cumulative Distribution Function (CDF) of a normal distribution. This is the sum of all the relative probabilities of the PDF, which is used to normalize the relative likelihood of a measurement to get its absolute probability. These equations are implemented in our MATLAB function phit.m, which is included in the Appendix.

IV. GPS Data

We used the Forward Equirectangular Projection to transform the GPS latitude and longitude measurements from the azimuth=0° file to X and Y coordinates in meters. We defined the origin to be located at the mean longitude and mean latitude measurements. The X-axis faces directly east, and the Y-axis faces directly north. See Figure 5.

We see that the variance in the X values is higher than in the Y values as there is a much larger spread in the X values than the Y values around the origin (the expected value). We can verify this statistically using MATLAB.

$$\Sigma = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} \\ \sigma_{yx} & \sigma_{yy} \end{bmatrix} = \begin{bmatrix} 5.0020 & -1.7651 \\ -1.7651 & 1.0143 \end{bmatrix}$$

V. Baye's Rule

We are given position states $\mathbf{x_1} = [-5.5\ 0]$, $\mathbf{x_2} = [5.5\ 0]$, and the probability of being in each state, $p(\mathbf{x_1}) = 0.5$, and $p(\mathbf{x_2}) = 0.5$. We want to calculate the probability of being in position states 1 and 2 given a measurement z: $p(\mathbf{x_1}|z)$ and $p(\mathbf{x_2}|z)$. We may find these probabilities using Baye's rule:

$$p(\mathbf{x}|z) = \frac{p(z|\mathbf{x})p(\mathbf{x})}{p(z)}$$

We can compute $p(z|\mathbf{x})$ from our sensor model, which we found to be a Gaussian. We are given $p(\mathbf{x_i})$, so all that remains is to calculate $p(z)$ using the Theorem of Total Probability. We know that the sum of all probabilities must equal 1:

$$1 = \sum_{i=1}^{2} p(\mathbf{x_i}|z)$$

$$1 = \sum_{i=1}^{2} \frac{p(z|\mathbf{x_i})p(\mathbf{x_i})}{p(z)}$$

$$p(z) = \sum_{i=1}^{2} p(z|\mathbf{x_i})p(\mathbf{x_i})$$

Thus, we can express $p(z)$ in terms of $p(z|\mathbf{x_i})$ and $p(\mathbf{x_i})$, which are both known quantities. Thus, $p(\mathbf{x}|z)$ is

$$p(\mathbf{x_j}|z) = \frac{p(z|\mathbf{x_j})p(\mathbf{x_j})}{\sum_{i=1}^{2} p(z|\mathbf{x_i})p(x_i)}$$

Using the $-90°$ azimuth range measurement, we find

$$p(\mathbf{x_1}|z) \approx p(\mathbf{x_2}|z) \approx 0.5$$

This makes sense given the values of $\mathbf{x_1}$, $\mathbf{x_2}$, and the $-90°$ azimuth range measurement. The $-90°$ azimuth range measurements vary from about $z = 9.25$m to $z = 9.31$m. Given that the sensors are in states $\mathbf{x_1} = [-5.5\ 0]$ or $\mathbf{x_2} = [5.5\ 0]$ and they would see a wall at $[11, 0]$, we would expect their range measurements to be 16.5m and 5.5m respectively. This suggests that there is approximately 0 probability that the sensor is in position states $\mathbf{x_1} = [-5.5\ 0]$ or $\mathbf{x_2} = [5.5\ 0]$. But

because the sensor must be in either state, we end up with a 50% probability for each. To illustrate this, we can plug in the numbers.

$$p(\mathbf{x_1}|z) = \frac{p(z|\mathbf{x_1})p(\mathbf{x_1})}{p(z|\mathbf{x_1})p(\mathbf{x_1}) + p(z|\mathbf{x_2})p(\mathbf{x_2})}$$
$$= \frac{p(z|\mathbf{x_1})0.5}{p(z|\mathbf{x_1})0.5 + p(z|\mathbf{x_2})0.5}$$

We know that $p(z|\mathbf{x_1}) \approx p(z|\mathbf{x_2}) \approx 0$. However, when we plug in 0, we get 0/0 so we must instead approximate $p(z|\mathbf{x_1}) \approx p(z|\mathbf{x_2})$ as a very small number. In our MATLAB code, we did this with an if statement. Using this approximation, we find that $p(\mathbf{x_1}|z) = p(\mathbf{x_2}|z) = 0.5$. This result makes sense. Our new measurement is incredibly unlikely, so it provides no information which can be used to improve the estimate of the sensor state, and we are left with the original probabilities.

APPENDIX A
MATLAB CODE

A. E205_Lab1.m

```
%% Lab 1 E205
%pjohnson@g.hmc.edu and pking@g.hmc.edu

%% 1 Read the Data
N90 = readtable('lab1_azimuth_-90.csv');
P00 = readtable('lab1_azimuth_00.csv');
P90 = readtable('lab1_azimuth_90.csv');
%% 2 Histograms
% Histograms of range data
% Determined bin size to appear the most normal
figure(1)
histogram(N90.Range_m_, 'BinWidth', .00375);
title('Histogram of range for azimuth = -90^{\circ}')
xlabel('Range[m]')
ylabel('Frequency')

figure(2)
histogram(P00.Range_m_, 'BinWidth', .00375);
title('Histogram of range for azimuth = 0^{\circ}')
xlabel('Range[m]')
ylabel('Frequency')

figure(3)
histogram(P90.Range_m_, 'BinWidth', .02);
title('Histogram of range for azimuth = 90^{\circ}')
xlabel('Range[m]')
ylabel('Frequency')


%% 4 Create a Model
%Use beam finder model
%Gaussian and uniform distribution
%Assume ground truth (mu) is average of the data, i.e. our data is not
%biased

pd = fitdist(N90.Range_m_,'Normal');

x = 9.25:0.0001:9.31; %This is just based on looking at the range of our data
y = pdf(pd,x);
figure(4)
plot(x,y,'LineWidth',2)

hold on
%Overlay a normalized histrogram to compare to normal pdf
histogram(N90.Range_m_, 'BinWidth', .00375, 'Normalization', 'pdf'); %
title('Normalized Histogram Fitted with Normal Distribution (azimuth = -90{\circ})')
xlabel('Range[m]')
ylabel('Probability')

% Now need to develop function to get probability
% equation for P(Z|X)
% beam range finder model
sigma_hit = pd.sigma;   %Variance
mu_hit = pd.mu;         %Expected value
```

```
zt_star = mu_hit;       %True value
zt = N90.Range_m_;      %Measurement
zmax_range = 100;       %max range is 100 [m] from datasheet

%See phit function for implementation

%% 5 Transform and Plot the GPS measurements
%use Azimuth=0 file
%Convert to radians for unit purposes
lats = deg2rad(P00.Latitude);
lons = deg2rad(P00.Longitude);

%Origin is average of measurements
orig_lat = mean(lats);
orig_lon = mean(lons);

%Transform to X and Y [m] coordinate grid
%Use Equirectangular projection
R_earth = 6.371*10^6; %[m]

X = R_earth*(lons-orig_lon)*cos(orig_lat);
Y = R_earth*(lats-orig_lat);

figure(5)
scatter(X,Y)
title('XY Coordinate representation of GPS');
xlabel('X [m]')
ylabel('Y [m]')
xlim([-5 5]);
ylim([-5 5]);

%From the graph, notice that there is muchmore variance in the X data

%Look at covariance to verify this
covXY = cov(X,Y);

%% 6 Implement Baye's Filter
%Given states
x1 = [-5.5,0.0];
x2 = [5.5,0.0]; %x(2,:)
%And probabilities of each state
px1 = 0.5;
px2 = 0.5;
%And a measurement
ztk = N90.Range_m_(1);
%Calculate Probability of being in each state using Baye's rule
% p(x|z) = p(z|x)*p(x)/p(z)

%Will first need p(z).
%From the notes, we have
%p(z) = sum( p(z|x=i)*p(x=i)

%Define true distances based on beam casting
ztk_star1 = 5.5+11;
ztk_star2 = 11-5.5;
%Get Conditional measurement probabilities
pzx1 = phit(ztk,sigma_hit, ztk_star1,zmax_range);
pzx2 = phit(ztk,sigma_hit, ztk_star2,zmax_range);
```

```
%Get measurement probability as normalizer
%Question about how to calculate pz
pz = sum(pzx1*px1+pzx2*px2); %So p(z) the normalizer is small
%Plug into baye's rule
px1z = pzx1*px1/pz;
px2z = pzx2*px2/pz;

%Added something to phit to prevent a value of 0 in pz
%Since there are only two locations where our sensor could be and they are
%equally unlikely, then it makes sense there is a 50% probability for each
%location
```

B. phit.m

```
function p = phit(ztk, sigma_hit, ztk_star, zmax) %dont take in xt or m,
%PHIT Find the probability of a good measurement
%   Correct range with local measurement noise measured as a gaussian
%   Instead of ray casting with a map, we take in a known true measurement
    if (0 <= ztk && ztk <= zmax)                        %Measurment is in bounds
        N = normpdf(ztk, ztk_star, sigma_hit);          % eq(6.4) pdf is a relative
    probability
        eta = normcdf([0 zmax], ztk_star, sigma_hit); % eq (6.6)
        eta = (eta(2)-eta(1))^-1;                       %normcdf is integral of pdf
    over the bounds
        p = N*eta;                                      %cdf is used to normalize
    probability
        %When probabilities are small, want to ensure they are nonzero
        if p == 0
            p=0.0001;
        end
    else
        p = 0;
    end
end
```