CS221 Programming Assignment 4 README

Anand Madhavan - 05401919 <manand@stanford.edu>
Elizabeth Lingg - 05215856 <elingg@stanford.edu>
Alec Go - 005516471 <alecmgo@stanford.edu>

Custom Sensor Model

Our sensor achieved an average error of **1.562994** (vs 3.372725 from the provided model).

The custom sensor model is a mixture model of the following distributions:

1. Local noise. This was represented by a Gaussian distribution with the mean centered at the true distance. This represents noise in the sensor model. The standard deviation was kept at 2.5 (same as in the original sensor).

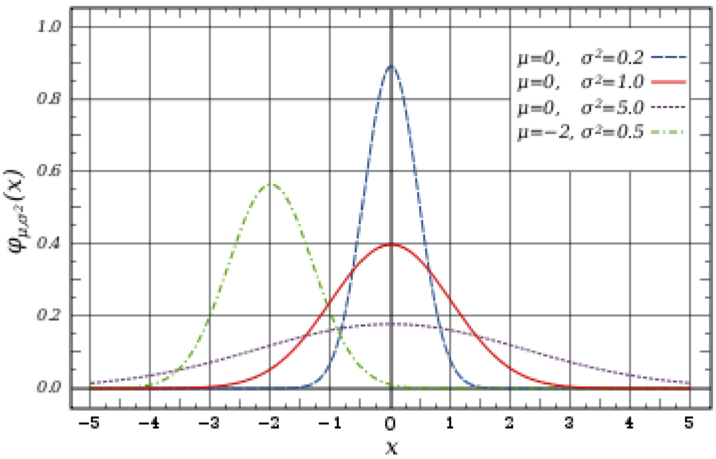


Figure 1. Image of Gaussian distribution from [1] Wikipedia

2. Unexpected obstacles that give unexpected "short" readings. This was represented by an exponential distribution. This distribution was chosen because it assigns more probability closer to the sensor and progressively less as the distance approaches the obstacle's true distance.

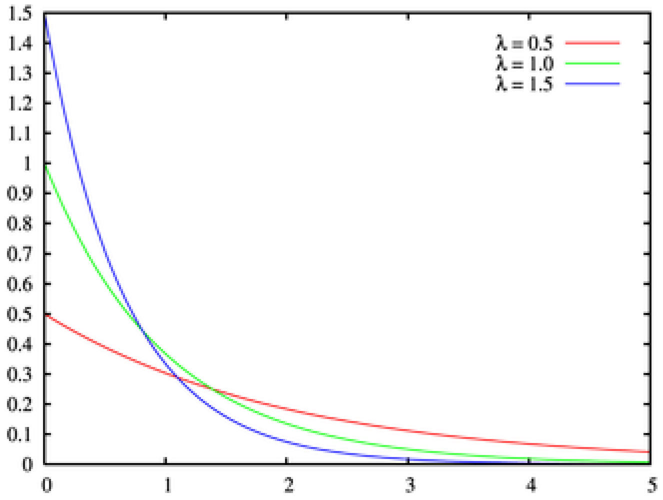


Figure 2. Image of Exponential Distribution from [2] Wikipedia

3. Laser failures caused by weird surfaces, bright light, etc. or obstacles outside the maximum range of the laser sensor.

This was represented by a Dirac delta distribution:

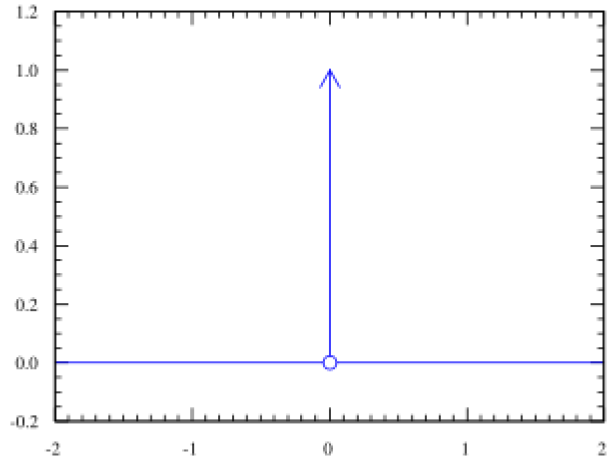


Figure 3. Image of Dirac Delta distribution from [3] Wikipedia

For our specific problem we use:
p(x) = 1 if (sensor = 30)
p(x) = 0 otherwise

That is, if the laser is broken, then we know with probability 1 that the sensor will have a value of 30.

4. Random, inexplicable measurements which are equally likely everywhere.

This was represented by a uniform distribution. This is the best distribution for modeling an equal probability at every distance.

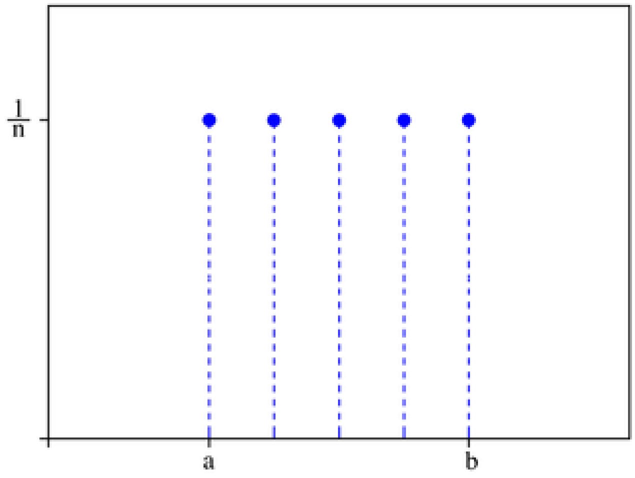


Figure 4. Image of Uniform Distribution from [4] Wikipedia

Choosing a Model Weights

We chose the following model weights:

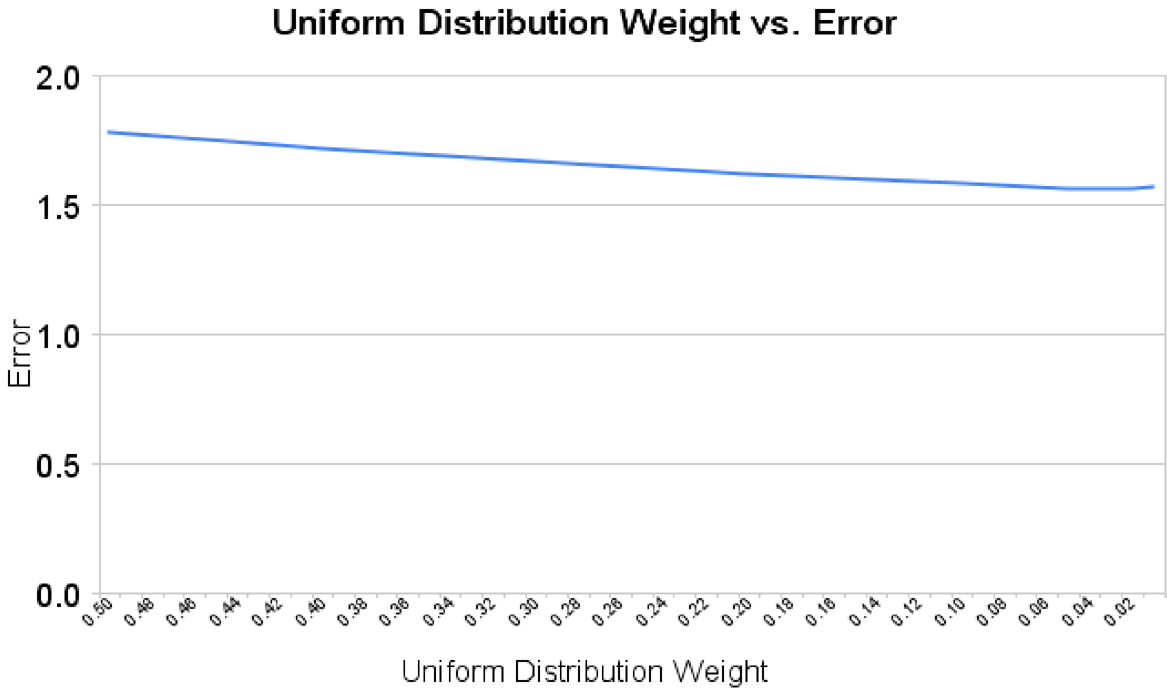
1. Local noise: **0.968**
2. Unexpected obstacles that give unexpected "short" readings: **0.001**

3. Laser failures caused by weird surfaces, bright light, etc. or obstacles outside the maximum range of the laser sensor: **0.001**

4. Random, inexplicable measurements which are equally likely everywhere: **0.03**

Through several experiments, we found that the local noise model (1) was the most significant in lowering error. The uniform distribution (2) had the second most impact. The other two distributions did very little for lowering the error, so these were assigned weights of 0.001.

The approach for choosing the model was to vary one weight to find the minimum while keeping all other weights static. After this minimum was found, another weight was varied to find the minimum. We only had to vary 3 out of the 4 weights, because the 4 weights sum up to 1. In the specific case of convex functions, this algorithm converges to the minimum. The following graph shows this approach taken, where the uniform distribution weight was varied. The minimum was found to be weight = 0.03.



Summing to One

Normal Distribution: The probability mass function returned a probability of zero for any negative sensor values. We ensured that the cumulative normal distribution summed to 1 by redistributing the "negative" portion throughout the rest of the probability function. The following formula was used for rebalancing:

```
double sensorAlpha = 1.0 / (1.0 - GetCND(0.0, distance, stddev));
double sensorModel = sensorAlpha * GetProbND(sensor, distance, stddev);
```

Mixed Model: There were 4 different weights, corresponding to each of the 4 distributions above. We ensured that the weights always added up to 1, so that the probabilitities of the mixed model also summed up to one.

Justification for the model we have chosen

We chose our final model based on the experiments (see sections above) we ran that gave the best average error (we introduced command line arguments for the weights that we can tune from the shell).
To summarize we use the following model:
a) Local noise model: A gaussian distribution with mean centered around the true distance, and a weight of 0.968
b) Unexpected obstacles model: An exponential distribution with decay factor of 0.5, and a weight of 0.001
c) Laser failure model: A Dirac delta distribution (that gives probability 1.0 for distances above 30m), and a weight of 0.001
d) Random, inexplicable measurements model: A uniform distribution, and a weight of 0.03

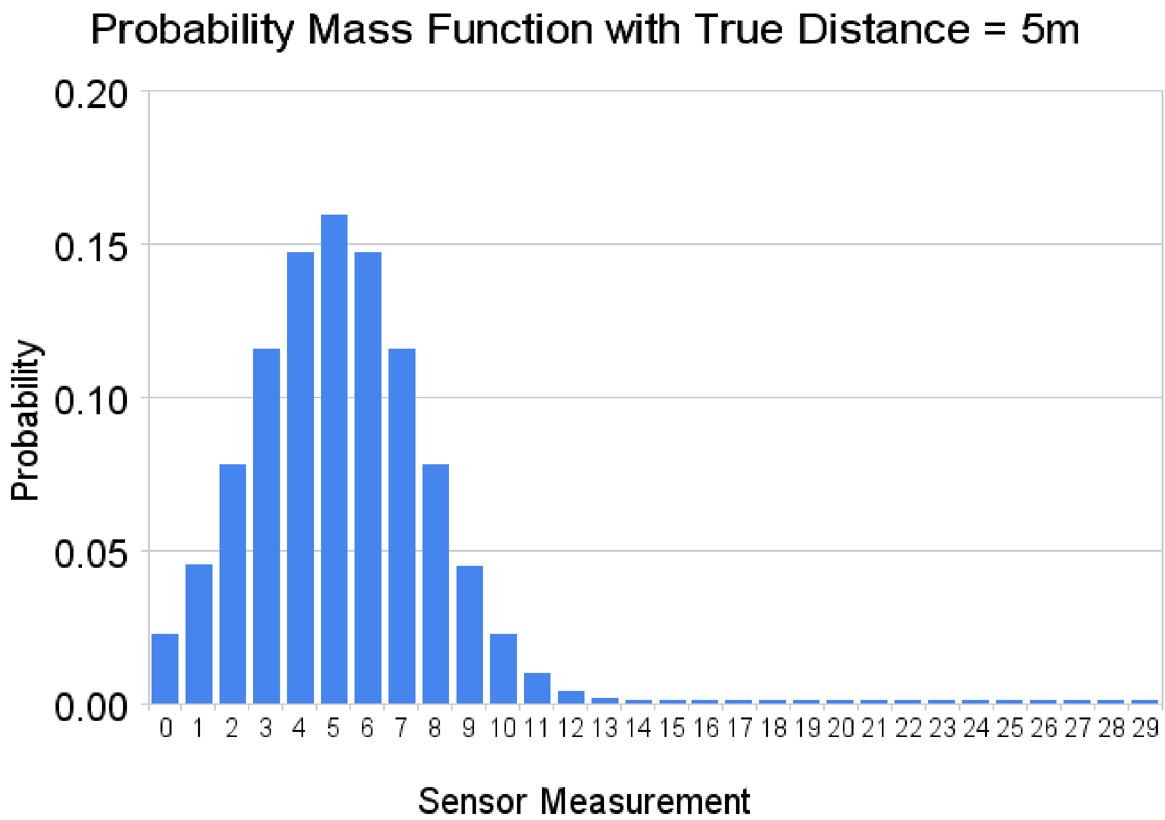
e) A combined mixture of the above model was used with the indicated weights to model the final custom sensor.

Results of the experiments we ran

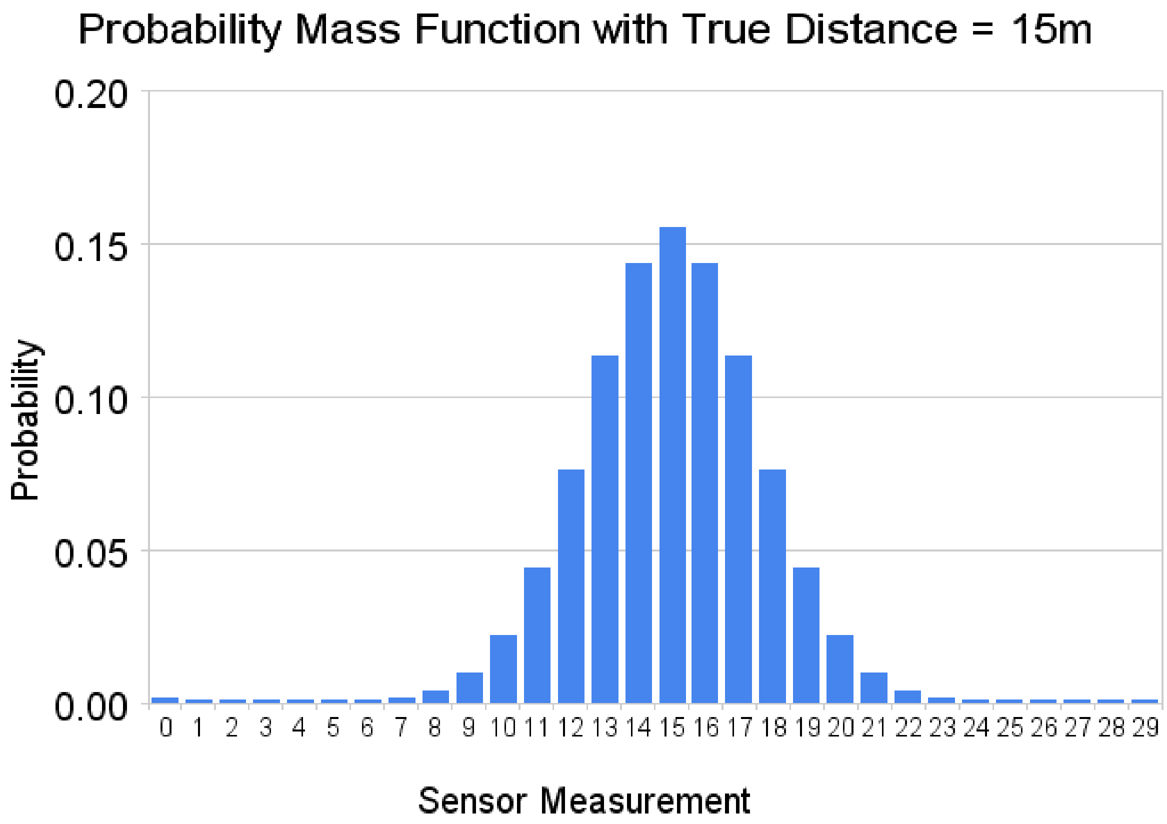
See section on 'Choosing Model Weights' above. In effect we kind of coordinate descent algorithm to find the best combination of weights that would minimize the error we got. We assume that this is a convex function and so we vary each parameter keeping the other fixed and finding the minimum in that direction and then proceeding to tweak the next parameter holding the others fixed and so on. See above for the chart/result of our experiments.

Graph of our sensor model's distribution

The following graph shows the probability mass function of our sensor model, with the true distance of = 5m.



The following graph shows the probability mass function with the true distance of = 15m.



Note: The normal distribution in the first graph is "taller" than in the second graph. The first graph has a peak of P(x) = 0.159108, and the second graph has a peak of P(x) = 0.155471. This is because the "negative" portion of the probabilities were distributed into the rest of the distribution.

References

[1] http://en.wikipedia.org/wiki/Normal_distribution
[2] http://en.wikipedia.org/wiki/Exponential_distribution
[3] http://en.wikipedia.org/wiki/Dirac_delta_function
[4] http://en.wikipedia.org/wiki/Uniform_distribution_(discrete)