# CS225b Final Project: SLAM

Drew Schmitt, Ruslan Kurdyumov

December 15, 2011

*In this project, we were tasked with designing a Simultaneously Localization and Mapping (SLAM) algorithm. Methods were derived from in-class instruction and the class-assigned textbook. Experimental methods using the simulator, rosbag files, and the actual robot were used to test the efficacy of the designed SLAM algorithm.*

## 1 Incremental Pose Estimation

Due to the fact that our team only had two members on the previous project, we were unable to write our own scan matching software. Thefore, for the first portion of this assignment, the Karto scan matching library was enlisted to perform the scan matching needed for incremental pose estimation. The Karto scan matcher takes a LaserScan, an estimated pose associated with that laser scan, and a set of reference laser scans and their associated poses. The estimated pose was determined using pure odometry measurements from the previous time the scan matcher was called. The reference scans were chosen to be the previous $K$ scans received. A value of $K = 4$ performed well in these experiments.

The Karto scan matcher returns a struct containing a corrected pose, a covariance matrix, and a reponse value. The corrected pose is Karto's pose guess of the actual location of the estimated pose passed in using pure odometry. The covariance matrix indicates the certainty of the corrected pose and is found by

$$\Sigma_{scan} = \begin{bmatrix} xx & yx & 0 \\ xy & yy & 0 \\ 0 & 0 & \theta\theta \end{bmatrix} \tag{1}$$

where the $xx$, $yy$, $\theta\theta$ are indicative of the uncertainty in the $x$, $y$, and $\theta$ directions, respectively. An example situation where the $yy$ term would be much different in magnitude than the $xx$ is when the robot is in a hallway and feels confident in localizing itself in one dimension. The cross terms $xy$ and $yx$ indicate the uncertainty along the diagonals between $x$ and $y$. The other terms in this covariance matrix are set to be zero because they have been found by experiment to not affect the outcome very much.

The reponse value returned by the Karto scan matcher indicates the amount of overlap between the new scan and the reference scans, which ranges between $0 \leq \text{response} \leq 1$. A small value for response indicates that there is very little overlap between the new scan and the reference scans so the values for the estimated pose and covariance matrix should not be fully trusted.

## 2 Graph Structure

Each new laser scan and pose that was received was stored into an internal graph structure for later reference. A set of parallel STL vectors was used to store the poses, laser scans, and barycenters of the associated laser scans. A separate class called $GraphManager$ was created to manage the adding of nodes, adding of constraints, optimizing, and visualizing.

To limit the amount of processing during lulls in activity, the rate of processing of laser scans was throttled by the amount the robot had moved since the previous processed laser scan. For example, if the robot received a laser scan and then sat motionless for awhile, no laser scans or nodes would be processed until the robot proceeded to move again.

The $addNode()$ function pushed back the scan matcher's corrected pose onto the list of visited nodes. The laser scan associated with this pose was pushed onto the vector of laser scans. The barycenter of this laser scan was automatically calculated using the mean in the $x$ and $y$ directions and then this value was pushed back onto the vector of barycenters.

To optimize these poses using $g2o$ (as described in the next section), edge constraints must be created and used to connect the poses. Henceforth, the "child" node will refer to the endpoint of the edge constraint and the "parent" node will refer to the origin of the edge constraint. A function called $addConstraint()$ requires the child's pose in the global frame and the covariance associated with this pose in the global frame. The covariance referenced here is shown above in Equation 1. Internally, this function converted the provided mean and covariance estimates out of the global frame and into the parent's frame. The mean in the parent's frame was calculated as

$$\mu_{parent} = \left\{T_W^P\right\}^{-1} \mu_{global} \tag{2}$$

where $T_W^P$ represents the parent's pose in the global frame and can be computed using SE2's built-in minus operator. The symbol $\mu_{global}$ is the child's pose in the global frame. In Equation 2, it is equivalent to treat $T_W^P$ as a transformation instead of a pose.

The first constraint passed into $addConstraint()$ was the scan matching constraint created using Karto. The child's mean and covariance in the global frame were computed directly using Karto and passed into this function. The second constraint created was the odometry constraint derived from the robot's motion model. The child's mean used in this case was the offset using pure odometry from the known parent's position, which is assumed to be accurate. The variables $\Delta x$, $\Delta y$, and $\Delta\theta$ were tracked since when the parent's pose was created. Therefore, the child's mean pose can be computed as

$$\begin{aligned}
\mu_{x,child,global} &= \mu_{x,parent,global} + \Delta x && (3)\\
\mu_{y,child,global} &= \mu_{y,parent,global} + \Delta y && (4)\\
\mu_{\theta,child,global} &= \mu_{\theta,parent,global} + \Delta\theta. && (5)
\end{aligned}$$

The covariance for odometry was found using the following model

$$\Sigma_{odom} = \begin{bmatrix} \cos\theta_p & -\sin\theta_p & 0 \\ \sin\theta_p & \cos\theta_p & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \Sigma_{xy,00} & \Sigma_{xy,01} & 0 \\ \Sigma_{xy,10} & \Sigma_{xy,11} & 0 \\ 0 & 0 & \sigma_{\theta\theta} \end{bmatrix} \begin{bmatrix} \cos\theta_p & -\sin\theta_p & 0 \\ \sin\theta_p & \cos\theta_p & 0 \\ 0 & 0 & 1 \end{bmatrix}^T, \tag{6}$$

where $\theta_p$ is the parent's pose in the global frame. The variable $\sigma_{\theta\theta}$ is a measure of the odometry's covariance when turning and is computed as

$$\sigma_{\theta\theta} = \alpha_4 \text{ abs}(\theta_p + \theta_c - 2\gamma) + \alpha_5 \delta_{trans}, \tag{7}$$

where $\alpha_4$ and $\alpha_5$ are user-selected parameters that control the odometry's error in angle per total angle and error in angle per total translation, respectively. Values of $\alpha_4 = \frac{1}{10}$ and $\alpha_5 = \frac{\pi}{180}$ were selected for this experiment. The variable $\theta_c$ is the child pose's angle in the global frame and $\gamma$ is the angle of travel between the parent's pose and the child's pose and can be calculated by

$$\gamma = \text{atan2}(\Delta x, \Delta y). \tag{8}$$

The last term in Equation 6 is the matrix containing the $xy$ covariance information and is found by

$$\Sigma_{xy} = \begin{bmatrix} \sigma_{xx} & 0 \\ 0 & \sigma_{yy} \end{bmatrix} \tag{9}$$

where the $\sigma_{xx}$ and $\sigma_{yy}$ terms are found by

$$\sigma_{xx} = \alpha_1 \delta_{trans} + \alpha_3 \text{abs}(\theta_p - \gamma) \tag{10}$$

$$\sigma_{yy} = \alpha_2 \delta_{trans} + \alpha_3 \text{abs}(\theta_p - \gamma) \tag{11}$$

where $\alpha_1$ and $\alpha_2$ represent the error in translational overshoot per total translation and error in translational drift per total translation, respectively. In this experiment, $\alpha_1 = 0.01$ and $\alpha_2 = 0.03$ seemed to model the robot's motion effectively.

## 3 Optimization

The optimization stage of this experiment simply involved installing the $g2o$ package and setting up the vertices and edges properly. The edges of $g2o$ correspond to the poses in the $GraphManager$ and the edges correspond to the constraints linking two poses (a child and a parent) in the $GraphManager$.

The $optimize()$ command was issued every time a new pose was received. The performance never appeared to slow down in our test cases. However, if there were thousands of vertices and constraints in the system, one might consider only calling $optimize()$ periodically.

## 4 Occupancy Grid

Ray tracing and probabilistic methods were used to reconstruct an occupancy grid from the laser scans of the optimized graph. First, every pose in the graph was considered to be the origin of a laser beam. Then, the endpoint of every laser scan at that pose was projected into the global frame and was considered to be the destination of the laser beam. The distance of beam $i$ is denoted as $z_i$. The global coordinate space in between the origin and destination was discretized into a cell width of 10 cm.

A simple ray tracing algorithm was used to determine which cells were passed through by the laser beam from the origin to the destination cells. The amount that the laser beam passed through the cell was neglected for simplicity and efficient of calculation. All of the cell indices that were visited were then computed back to their global coordinate equivalents. The Euclidean distance between the origin and visited cell $j$ was found as

$$m = ||cell_{origin} - cell_j||_2. \tag{12}$$

Two piecewise distributions were used to represent $P(z|m,x)$ and $P(z|\neg m,x)$. Figure 1 shows the distribution when an obstacle is assumed to be present at $m$ and Figure 2 shows the distribution when an obstacle is assumed to not be present at $m$. Note that Figure 1 and 2 are represented on different vertical scales.

The $P(z|m,x)$ distribution consists of three piecewise components: uniform region if $m > z$, a Gaussian region if $m \approx z$, and a max range region if $z = \text{range\_max}$. The $P(z|\neg m,x)$ distribution consists of two piecewise components: uniform region if $z \neq \text{range\_max}$ and a max range region if $z = \text{range\_max}$. The scaling for these regions in $P(z|m,x)$ and $P(z|\neg m,x)$ were chosen as described in lecture. The only parameter worth discussing further is the $\sigma$ value of the Gaussian region in $P(z|m,x)$. Many different values of $\sigma$ were experimented with but it was experimentally determined that a value of

$$\sigma = \frac{\text{range\_max}}{300}, \tag{13}$$

produced the highest quality Occupancy Grid output. Finally, the log of the ratio between $P(z|m,x)$ and $P(z|\neg m,x)$ and is denoted as $l(i)$. This value is totaled for each cell $k$ for each laser scan $j$ at each pose $i$ as

$$l(i) += \log\left(\frac{P(z|m,x)}{P(z|\neg m,x)}\right) \tag{14}$$

Then, it is necessary to loop back through each cell and threshold its value of $l(i)$ to determine if it is a wall point or not. In this experiment, it was found that a threshold value of

$$\text{wall\_cutoff} = \text{NUMBER\_WALL\_HITS}\ \log(\text{oddsOfExactHit}) \tag{15}$$

where NUMBER_WALL_HITS is the number of times this exact location has to be hit by a laser to consider it an actual wall. And oddsOfExactHit is the maximum

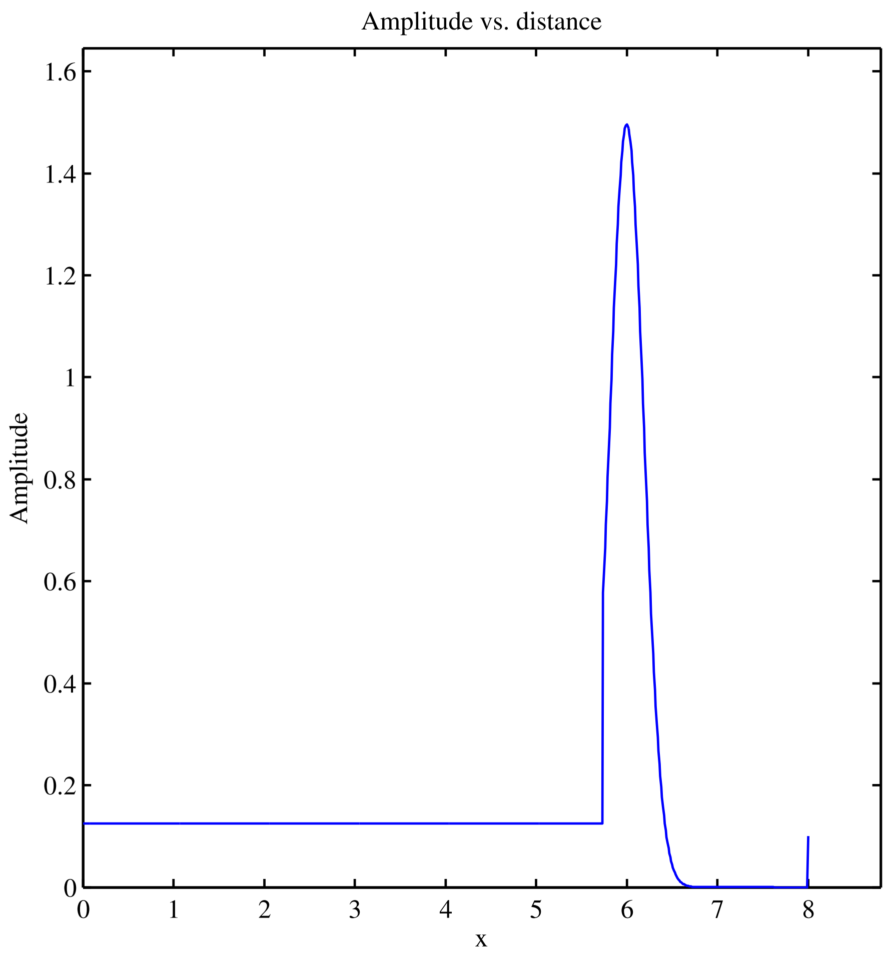

Figure 1: Distribution given $m$ assuming there is an obstacle present, $P(z|m,x)$.

value produced by the Gaussian region of the $P(z|m,x)$ distribution, which denotes an exact hit.

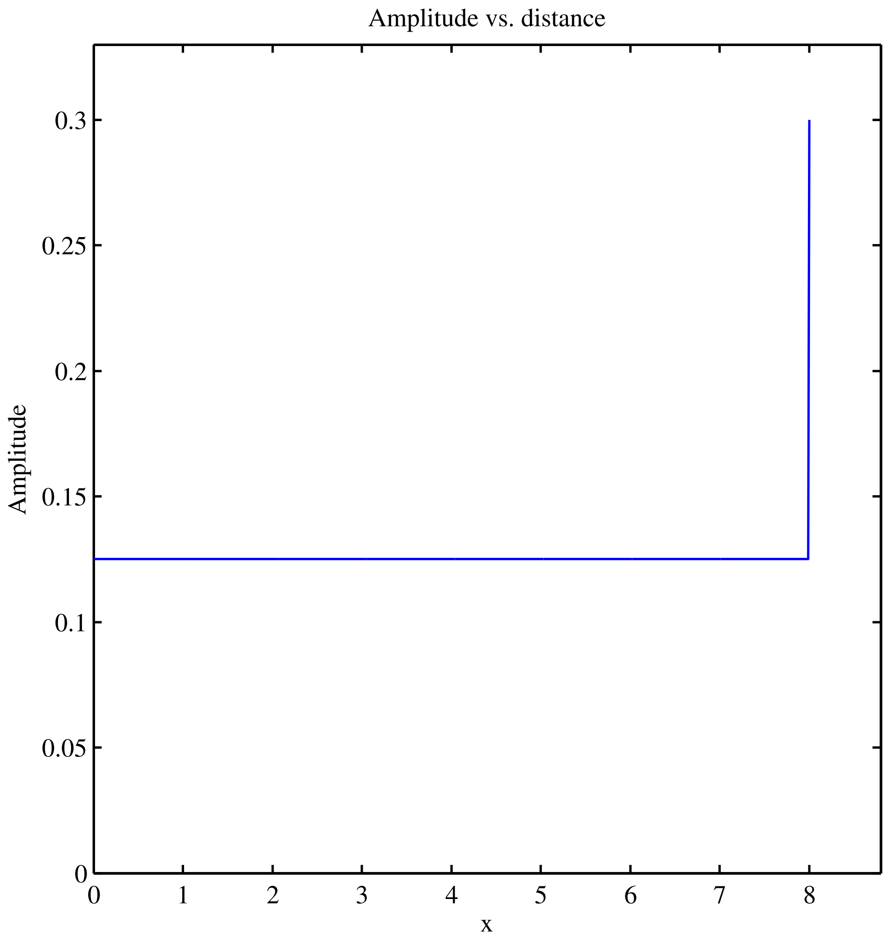

Figure 2: Distribution given $m$ assuming there is no obstacle present, $P(z|\neg m, x)$.