Nonlinear Constraints

In the Localization lesson, you were introduced to nonlinear motion and measurement models. The idea that a robot only moves in a linear fashion is quite limiting, and so it became important to understand how to work with nonlinear models. In localization, nonlinear models couldn't be applied directly, as they would have turned the Gaussian distribution into a much more complicated distribution that could not be computed in closed form (analytically, in a finite number of steps). The same is true of nonlinear models in SLAM - most motion and measurement constraints are nonlinear, and must be linearized before they can be added to the information matrix and information vector. Otherwise, it would be impractical, if not impossible, to solve the system of equations analytically.

Luckily, you will be able to apply the same procedure that you learned in the EKF lesson of Localization to linearize nonlinear constraints for SLAM.

If you recall, a Taylor Series approximates a function using the sum of an infinite number of terms. A linear approximation can be computed by using only the first two terms and ignoring all higher order terms. In multi-dimensional models, the first derivative is replaced by a Jacobian - a matrix of partial derivatives.

Linearizing Constraints

A linearization of the measurement and motion constraints is the following,

$$g(u_t, x_{t-1}) \simeq g(u_t, \mu_{t-1}) + G_t(x_{t-1} - \mu_{t-1})$$

$$h(y_t) \simeq h(\mu_t) + H_t^i(y_t - \mu_t)$$

To linearize each constraint, you need a value for μ_{t-1} or μ_t to linearize about. This value is quite important since the linearization of a nonlinear function can change significantly depending on which value you choose to do so about. So, what μ_{t-1} or μ_t is a reasonable estimate for each constraint?

Well, when presented with a completed graph of nonlinear constraints, you can apply only the motion constraints to create a pose estimate, $[x_0 \ldots x_t]^T$, and use this primitive estimate in place of μ to linearize all of the constraints. Then, once all of the constraints are linearized and added to the matrix and vector, a solution can be computed as before, using $\mu = \Omega^{-1}\xi$.

This solution is unlikely to be an accurate solution. The pose vector used for linearization will be erroneous, since applying just the motion constraints will lead to a graph with a lot of drift, as errors accumulate with every motion. Errors in this initial pose vector will propagate through the calculations and affect the accuracy of the end result. This is especially so because the errors may increase in magnitude significantly during a poorly positioned linearization (where the estimated μ_t is far from reality, or the estimated μ_t lies on a curve where a small step in either direction will make a big difference).

To reduce this error, we can repeat the linearization process several times, each time using a better and better estimate to linearize the constraints about.

Iterative Optimization

The first iteration will see the constraints linearized about the pose estimate created using solely motion constraints. Then, the system of equations will be solved to produce a solution, μ.

The next iteration will use this solution, μ, as the estimate used to linearize about. The thought is that this estimate would be a little bit better than the previous; after all, it takes into account the measurement constraints too.

This process continues, with all consequent iterations using the previous solution as the vector of poses to linearize the constraints about. Each solution incrementally improves on the previous, and after some number of iterations the solution converges.

Summary

Nonlinear constraints can be linearized using Taylor Series, but this inevitably introduces some error. To reduce this error, the linearization of every constraint must occur as close as possible to the true location of the pose or measurement relating to the constraint. To accomplish this, an iterative solution is used, where the point of linearization is improved with every iteration. After several iterations, the result, μ, becomes a much more reasonable estimate for the true locations of all robot poses and features.

The workflow for GraphSLAM with nonlinear constraints is summarized below:

- Collect data, create graph of constraints,
- Until convergence:
  - Linearize all constraints about an estimate, μ, and add linearized constraints to the information matrix & vector,
  - Solve system of equations using $\mu = \Omega^{-1}\xi$.

Search or ask questions in Knowledge.

Ask peers or mentors for help in Student Hub.