Incremental Optimization of Large Robot-Acquired Maps

Edwin Olson EOLSON@MIT.EDU
MIT Computer Science and Artificial Intelligence Laboratory, 32 Vassar Street, Cambridge MA, 02139 USA

1. Introduction

Many robotics applications require the robot to build a map of its environment. We consider the problem of *Simultaneous Localization and Mapping* (SLAM), i.e., building a map of an environment while exploring it for the first time. SLAM algorithms approach this by identifying features in the environment (e.g., the corner of a desk) and determining the relative positions of features. A robot's sensors are imperfect, so the relative position of one feature to another is almost always considered probabilistically– typically with a Gaussian distribution.

We can think about the map as a graph: features are nodes in the graph, and measurements which relate two features are edges. Each edge represents a rigid-body transformation and its uncertainty. We make the realistic assumption that each edge represents an *independent* constraint.

The heart of the SLAM problem is to determine the "best" map, the physical locations of features such that the constraints have maximum probability. We consider the case where the features are locations visited by the robot; as shown by (Montemerlo, 2003), positions of other features can be efficiently computed once the robot trajectory is known.

The classical method for SLAM problems is the Extended Kalman Filter (EKF). However, the EKF has several undesirable aspects: for N features it is $O(N^2)$ in space and time. It also performs poorly in the presence of highly non-linear constraints. The latter is true because the EKF commits to a linearization point at the time each constraint is incorporated; if the point at which the linearization occurred was inaccurate, linearization errors are introduced that cannot be undone later. In the SLAM problem, the orientation of the robot appears in most of the constraint equations in sine and cosine operations, which result in substantial linearization errors when the heading is not well-known. Sparse Extended Information Filters (SEIFs) also suffer from linearization errors, and incur $O(N^3)$ costs when computing the state estimate.

In this paper, we present an algorithm for optimizing pose graphs that is dramatically faster than the published state of the art. The improved performance arises from two sepa-

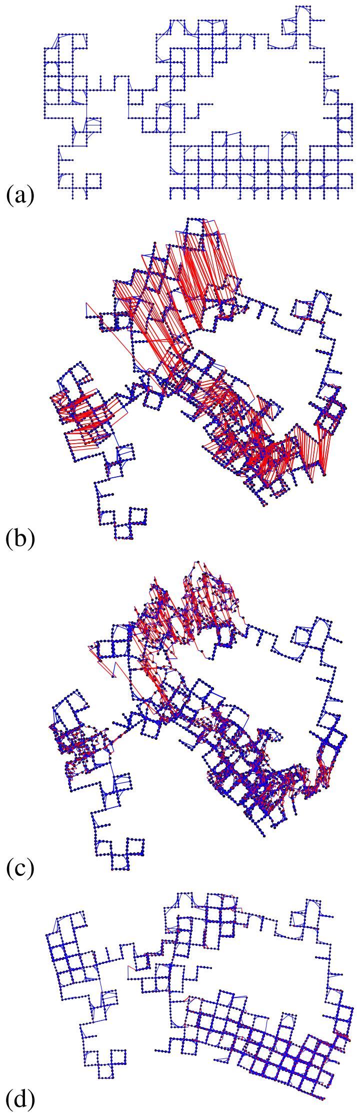

Figure 1. Sample synthetic problem. 3500 nodes with 2100 additional loop closures (A is 10500×10500, J is 16797×10500). Poses are shown as dots, lines represent constraints. Subplot (a) shows the ground truth map, (b) shows the simulated corrupted raw data, (c) shows the result of Duckett's Gauss-Siedel after 30 seconds of CPU time, and (d) shows the results of our method after convergence (about 10 seconds). The convergence rates for this experiment are shown in Fig. 3

rate ideas:

- The use of a different state representation which leads to a Jacobian that is better-suited to local iterative methods

- A variant of Stochastic Gradient Descent (SGD), a local iterative optimization method which escapes local minima more readily than Gradient Descent, Conjugate Gradient Descent, or Gauss-Seidel. Our variant exploits additional information available in the SLAM problem, allowing Newton steps rather than simple gradient steps.

2. Derivation

The maximum likelihood map can be incrementally computed using an iterative numerical approach. This approach has a number of distinct advantages: memory grows only as $O(N+E)$ (for N features and E edges), it can relinearize observations as the state estimate changes, and the incremental nature of the optimization means that an approximate map is always available for online path planning and exploration. The full state covariance never needs to be explicitly computed, however it can be reconstructed if necessary. A family of such approaches has been studied before (Duckett et al., 2000) and improved (Frese et al., 2005).

Before proceeding, we show how the graph can be optimized by solving a linear problem ($Ax = b$).

If x is the state vector representing robot poses, and $f()$ represents the constraint equations with expected values u and variances Σ, we can write:

$$-log\ P(x) \propto (f(x) - u)^T \Sigma^{-1} (f(x) - u) \qquad (1)$$

We proceed by linearizing $f(x) = F|_x + J|_x \Delta x$, using matrices $F|_x$ and $J|_x$. At any particular iteration, we will simply write F and J, and will use $d = \Delta x$. We also set $r = u - F$, the residual. Eqn 1 then becomes:

$$\begin{aligned} -log\ P(x) &\propto (Jd - r)^T \Sigma^{-1} (Jd - r) \\ &= d^T J^T \Sigma^{-1} J d - 2 d^T J^T \Sigma^{-1} r + r^T \Sigma^{-1} r \end{aligned}$$

We wish to improve our map by finding a d that maximizes the probability. Differentiating with respect to d and setting to zero, we find that:

$$(J^T \Sigma^{-1} J) d = J^T \Sigma^{-1} r \qquad (2)$$

This is the elementary $Ax = b$ linear algebra problem. If we solved for d directly (via inversion of A, or better by LU decomposition), we would have the method of nonlinear least squares. However, the size of A makes a direct solution impractical. Instead, we will estimate d.

When the state estimate is corrupted by significant noise, the local gradient will typically not point in the direction of the global minimum. Consequently, gradient methods typically fail to achieve a satisfactory solution.

In the SLAM problem, individual constraints all result in quadratic surfaces, ideal for optimization. It is only the sum of a number of constraints that leads to difficulties, so we propose using iterative methods that operate on only one constraint at a time. The optimal d can be written in terms of the sums of individual constraints by rewriting Eqn. 2 as:

$$d = (J^T \Sigma^{-1} J)^{-1} \sum J_i^T \Sigma_i^{-1} r_i \qquad (3)$$

Naturally, we still cannot invert the information matrix ($J^T \Sigma^{-1} J$), but we can approximate the inverse using its diagonal elements; this approximation preserves the local gradient of the cost function. This is roughly equivalent to Jacobi Preconditioning, which uses the same approximation.

The canonical Stochastic Gradient Descent algorithm iteratively evaluates the gradient for each constraint (one constraint per iteration) and moves x in the opposite direction at a rate proportional to the *learning rate*. In the SLAM context, we can do better; we know what step size corresponds to a *Newton step*– a step that obliterates the residual of a given constraint. We still employ a learning rate parameter in order to ensure convergence, but the Newton step serves as an upper bound. While extensive research has been done in the area of learning rate schedules, we have found that a simple harmonic series ($1/t$) as originally suggested by (Robbins & Monro, 1951) works well.

3. State Space Representation

Previous authors used the *absolute global position* for their state space; i.e., the state vector was composed of (x, y, θ) values. The Jacobian of a rigid-body constraint between two poses is consequently *sparse*, acting like a “spring” connecting just those two poses. However, in addition to a loop-closure constraint, there is a segment of the robot's path that connects any two poses. For example, in Fig. 2, a loop constraint exists between poses A and D, but there is an additional path between A and D that goes through poses B and C.

If we alter the relative alignment of poses A and D in order

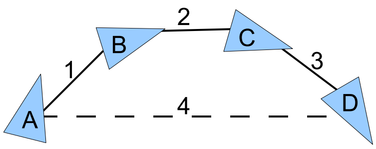

Figure 2. A simple pose graph. Optimizing constraint 4 typically has an effect on nodes B and C, in addition to A and D. Our proposed state space representation causes this dependency to appear in the Jacobian of constraint 4. This leads to more rapid convergence when using local iterative methods.

to reduce the error of constraint 4, poses B and C will also adjust position so that the total error will be reduced (due to the effects of constraints 1, 2, and 3.) Iterative methods, which use only a subset of the constraint information on each step, are unlikely to properly adjust B and C when they adjust constraint 4, since the effects of constraint 4 on nodes B and C appear in different rows of the Jacobian. This means that iterative updates to the state vector will be of poorer quality.

The Jacobian is a function of not just the constraint, but also the state space representation. If we change the state space representation, we can achieve a Jacobian that *does* capture the impact of moving two distant nodes on the nodes between them.

We use the *incremental global position* state space, in which the position of nodes is given relative to the previous node in the robot's trajectory, i.e.:

$$x = \begin{bmatrix} x_0 \\ y_0 \\ \theta_0 \\ x_1 - x_0 \\ y_1 - y_0 \\ \theta_1 - \theta_0 \\ \ldots \end{bmatrix} \qquad (4)$$

The relative position of two nodes is now a function of all the incremental positions between them, so each row of the Jacobian now incorporates "springs" for all of the intermediate nodes. When J is premultiplied by the approximate inverse of the information matrix (from Eqn. 3), the "stiffnesses" of the intermediate linkages is set according to the strength of all the constraints which involve each node.

4. Results

With the incremental state space representation, the Newtonized Stochastic Gradient Descent algorithm estimates search directions d much more effectively, leading to rapid convergence, as illustrated in Fig. 1. Gauss-Seidel converges much more slowly.

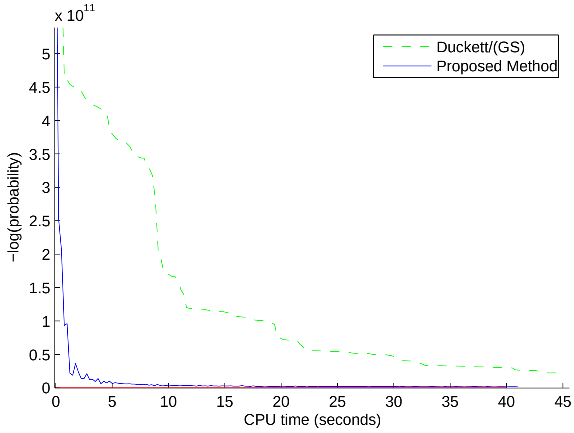

Figure 3. Convergence rates for Duckett's Gauss-Seidel approach and our method. Our method rapidly escapes local minima, converging quickly to a low-error solution.

Results of our algorithm are shown in Fig. 1. When the input graph is noisy, our method converges much more quickly than Gauss-Seidel relaxation, as shown in Fig. 3.

5. Conclusion

We have presented an iterative method for rapidly optimizing pose graphs, even in the presence of substantial initialization noise. This method shows promise in solving one of the open problems in SLAM: optimizing pose graphs after accumulating substantial error.

References

Duckett, T., Marsland, S., & Shapiro, J. (2000). Learning globally consistent maps by relaxation. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'2000)*. San Francisco, CA.

Frese, U., Larsson, P., & Duckett, T. (2005). A multilevel relaxation algorithm for simultaneous localisation and mapping. *IEEE Transactions on Robotics*.

Montemerlo, M. (2003). *FastSLAM: A factored solution to the simultaneous localization and mapping problem with unknown data association*. Doctoral dissertation, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

Robbins, H., & Monro, S. (1951). A stochastic approximation method. *Annals of Mathematical Statistics*, *22*, 400–407.