SLAM Strategy for an Autonomous Quadrotor

Pratik Agarwal
Computer Science and Engineering
University of Michigan
Ann Arbor, Michigan - 48105
Email: pratikag@umich.edu

Tom Brady
Aerospace Engineering
University of Michigan
Ann Arbor, Michigan - 48105
Email: bradthom@umich.edu

***Abstract*—This report presents a solution to the Simultaneous Localization and Mapping (SLAM) problem for a quadrotor in an office environment. We have implemented a robust, efficient Correlative Scan-Matcher[2] that is able to extract rigid body constraints between both successive and non-successive poses. These pose constraints are used to generate a map via an Exactly Sparse Delayed-State Filter[1]. We present our approach and follow with experimental results for simulated data of a robot driving through the streets of Manhattan and then for a real data set generated by a ground robot with a laser scanner on the A-level of Carnegie Melon University′s Newell-Simon Hall[5].**

Fig. 1. Image of the quadrotor

I. Introduction

Michigan Autonomous Aerial Vehicles (MAAV) was founded in the fall of 2009 with intent to compete in the International Aerial Robotics Competition (IARC). The IARC is an annual event hosted by the Association for Unmanned Vehicle Systems International (AUVSI). Teams from around the world have come to display cutting edge UAV technology at the competition for the past 20 years. The current mission (Mission 6) requires teams to infiltrate an unknown building that is part of a Nari Military Compound. Competitors are required to design a UAV capable of autonomous takeoff and undetected entry into the facility. Once inside the vehicle must follow Arabic signs leading it to the Chief of Security′s office where it will locate a small USB thumb drive that must be extracted from the building and replaced with a decoy. More information can be found at http://iarc.angel-strike.com/.

The competition environment is GPS-denied and thus requires teams to localize and map the environment in real time. The environment is also sparse of unique features, thus limiting the methods by which we can close loops in the map. This report presents our solution to the problem statement described above.
The two main parts of our system are:

- Simultaneous Localization and Mapping
- Scan-Matching

The quadrotor sensor payload includes a Hokuyo laser scanner and an inertial measurement unit. We have implemented a Correlative Scan-Matcher [2] that extracts rigid body constraints and their associated covariances from lidar returns. An Exactly Sparse Delayed-State Filter [1] makes use of these constraints to track the entire robot trajectory and the uncertainty associated with each pose in the map. Our system architecture is shown in Fig.2.

II. Review

A. Pose GraphSLAM

Simultaneous localization and mapping (SLAM) is a method to help robots explore, navigate, and map an

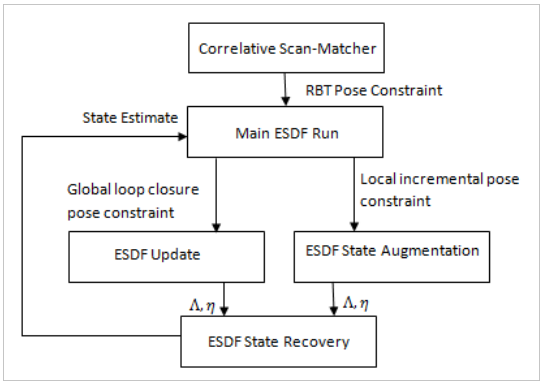

Fig. 2. Flowchart of the overall concept

unknown environment [6], [7]. It is well known that traditional methods for SLAM based on the extended Kalman filter (EKF) suffer computational complexity problems when dealing with large scale environments, as well as inconsistencies for non-linear SLAM problems. To incorporate non-linear measurements, current research has been focused on smoothing approaches using Graph SLAM techniques. The Exactly Sparse Delayed-State Filter (ESDF) was used by Eustice et al[1] to map the RMS titanic using pose constraints between camera views. We use the same formulation as in [1] as a starting point for our Graph SLAM implementation.

ESDF: The ESDF consists of four main parts; state augmentation, measurement updates, motion prediction, and state recovery.

State augmentation: At each time step, a new pose is added to the Markov chain, or the most recent pose is propagated in time via motion prediction. The posterior distribution is given by:

$$p(x_t, x_{t+1}, M|z^t, u^{t+1})$$

where x_t represents the current robot state and M is the map. The letter M is arbitrary as it may represent landmarks, poses, or a combination of the two. Factoring and applying markovity yields the new target distribution:

$$p(x_{t+1}|x_t, u_{t+1})p(x_t, M|z_t, u_t)$$

The robot′s state is assumed to evolve according to the model:

$$\begin{aligned} x_{t+1} &= f(x_t, u_{t+1}) + w_t \\ &\approx f(\mu_{x_t}, u_{t+1}) + F(x_t - \mu_{x_t}) + wt \end{aligned}$$

where F is the jacobian of the process model evaluated around the mean and w_t is zero-mean white process noise. Therefore the state representation is augmented as follows:

Covariance Form

$$\mu'_{t+1} = \begin{bmatrix} f(\mu_{x_t}, u_{t+1}) \\ \mu_{x_t} \\ \mu_M \end{bmatrix}$$

$$\Sigma'_{t+1} = \begin{bmatrix} (F\Sigma_{x_t x_t}F^T + Q) & F\Sigma_{x_t x_t} & F\Sigma_{x_t M} \\ \Sigma_{x_t x_t}F^T & \Sigma_{x_t x_t} & \Sigma_{x_t M} \\ \Sigma_{M x_t}F^T & \Sigma_{M x_t} & F\Sigma_{MM} \end{bmatrix}$$

InformationForm

$$\eta'_{t+1} = \begin{bmatrix} Q^{-1}(f(\mu_{x_t}, u_{t+1}) - F\mu_{x_t}) \\ \eta_{x_t} - F^T Q^{-1}(f(\mu_{x_t}, u_{t+1}) - F\mu_{x_t}) \\ \eta_M \end{bmatrix}$$

$$\Lambda'_{t+1} = \begin{bmatrix} Q^{-1} & -Q^{-1}F & 0 \\ -F^T Q^{-1} & \Lambda_{x_t x_t} + F^T Q^{-1} F & \Lambda_{x_t M} \\ 0 & \Lambda_{M x_t} & \Lambda_{MM} \end{bmatrix}$$

where Q is the covariance associated with the zero-mean Gaussian white process noise. It is to be noted that the lower left and upper right hand blocks of the information matrix are exactly zero.

Measurement Update: In the information form, measurement updates are constant time. A measurement in the ESDF framework corresponds to a pose-constraint between two poses x_i and x_j. The equations for measurement updates in the information and covariance form are shown below.

Covariance Form

$$\begin{aligned} K &= \bar{\Sigma}_t H^T (H\bar{\Sigma}_t H^T + R)^{-1} \\ \mu_t &= \bar{\mu}_t + K(z_t - h(\bar{\mu}_t)) \\ \Sigma_t &= (I - KH)\bar{\Sigma}_t (I - KH)^T + KRK^T \end{aligned}$$

Information Form

$$\begin{aligned} \eta_t &= \bar{\eta}_t + H^T R^{-1}(z_t - h(\bar{\mu}_t) + H\bar{\mu}_t) \\ \Lambda_t &= \bar{\Lambda}_t + H^T R^{-1} H \end{aligned}$$

Here K is the Kalman gain, R is the zero-mean Gaussian white observation noise, and H represents the Jacobian of the observation model with respect to the poses x_i and x_j.

$$H = \begin{bmatrix} 0 & \dots & \frac{\partial h}{x_i} & \dots & 0 & \dots & \frac{\partial h}{\partial x_i} & \dots & 0 \end{bmatrix}$$

In general, $H^T R^{-1} H$ is fully dense and therefore modifies all elements in the information matrix. However, in the ESDF framework, the H matrix is sparse and therefore measurement updates require only changing the blocks of the information matrix associated with the poses being observed.

Motion prediction: The robot′s state is propagated forward in time whenever a new pose is not being added to the state representation. This involves marginalizing out the previous state, x_t. Marginalization in the information form is generally computationally intense. However in the ESDF framework, measurement prediction is constant time because each pose only shares information with the pose directly before and after.

$$\bar{\eta}_{t+1} = \begin{bmatrix} Q^{-1}F\Omega^{-1}\eta_{x_t} + \psi(f(\mu_{x_t}, u_{t+1}) - F\mu_{x_t}) \\ \eta_M - \Lambda_{Mx_t}\Omega^{-1}\eta^{\star}_{x_t} \end{bmatrix}$$

$$\bar{\Lambda}_{t+1} = \begin{bmatrix} \psi & Q^{-1}F\Omega^{-1}\Lambda_{x_tM} \\ \Lambda_{x_tM}\Omega^{-1}F^TQ^{-1} & \Lambda_{MM} - \Lambda_{Mx_t}\Omega^{-1}\Lambda_{x_tM} \end{bmatrix}$$

$$\eta^{\star}_{x_t} = \eta_{x_t} - F^TQ^{-1}(f(\mu_{x_t}, u_{t+1}) - F\mu_{x_t})$$
$$\Omega = \Lambda_{x_tx_t} + F^TQ^{-1}F$$
$$\psi = (Q + F\Lambda^{-1}_{x_tx_t}F^T)^{-1}$$

State recovery: State recovery can be done naively by inverting the information matrix. This involves cubic complexity. However as the information matrix is sparse, state recovery may be performed by solving a linear system of equations with quadratic complexity.

$$\Lambda_t\mu_t = \eta_t$$

[1] pioneered a method to partially recover the robot state by partitioning the map into a set of local and benign states. Local states are in close proximity to the robot, whereas the benign states are not. This partitioning allows solving for the local portion of the map in constant-time.

$$\hat{\mu}_l = \Lambda^{-1}_{ll}(\eta_l - \Lambda_{lb}\hat{\mu}_b)$$

B. Scan-Matcher

Iterative Closest Point (ICP) [8], [9] and Iterative Closest Line (ICL) [10], [11], [12] are used commonly for scan matching. In ICP, each point in the query scan is associated with the reference scan according to a distance metric (most commonly Euclidean distance). A rigid-body transformation that best aligns the reference and query points can then be computed. Heuristics such as Hill-Climbing search methods are used to enhance computation. It is shown in [2] that as the initial estimate of the rigid-body transformation deteriorates, so does the quality of the output for both ICP and Hill-Climbing. In our case we are formulating the problem considering that we do not have any prior odometry measurements as the motion model of a quadrotor is commonly unreliable. We have chosen to implement scan matching using a Correlative Scan-Matcher to recover laser odometry.

Correlative Scan-Matcher: Correlative Scan Matching [2] is a relatively new approach to the scan matching problem that was developed by Professor Edwin Olson of the University of Michigan′s Computer Science department. It is a probabilistically motivated algorithm that produces higher quality and more robust results. Unlike ICP and Hill-Climbing techniques, Correlative Scan-Matching is not vulnerable to converge at local minima. The robustness is achieved at the cost of additional computational time. The algorithm may be implemented on a GPU to improve efficiency. It makes use of a multi-resolution look-up table that greatly accelerate the computations.The Correlation based Scan-Matching approach results in both a more robust maximum likelihood estimate and a principled estimation of uncertainty.

The problem is formulated such that the robot is moving from x_{i-1} to x_i , according to some motion u. The observation z is dependent on the environment model m and the robots position. Our goal is to find the posterior distribution over the robots position, $p(x_i|x_{i-1}, u, m, z)$. By applying Bayes rule and removing irrelevant conditionals, we get

$$p(x_i|x_{i-1}, u, m, z) \propto p(z|x_i, m)p(x_i|x_{x-1}, u) \quad (1)$$

The first term, $p(z|x_i, m)$ is the observation model: how likely is a particular observation, if the environment and the robots position are known? The second term, $p(x_i|x_{i-1}, u)$ is the motion model of the robot, as obtained (for example) from control inputs or odometry.

Each individual lidar return z_j is independent giving Eq.2:

$$p(z|x_i, m) = \prod_j p(z_j|x_i, m) \quad (2)$$

The computation of the probability $p(z|x_i, m)$ can be accelerated by building a 2D lookup table. The 2D table is a rasterized cost table containing log probabilities of lidar observation at each (x,y) position of the world. Here m is the previous scan that we are matching to. For large loop closures m is the model or the reference scan. In principle, we need to evaluate $p(z|x_i, m)$ over a three-dimensional volume of points; the three dimensions corresponding to the unknown parameters of the rigid body transformation, T : Δx,Δy, and $\Delta\theta$. This is a brute force method but can be sped up using certain techniques explained later.

One of the biggest advantages of a Correlation based Scan-Matcher is a principled evaluation of uncertainty covariances. Once the value of the cost function has been evaluated over a range of values of x_i , a multivariate Gaussian distribution can be fit to the data as per Eqn.3.

Let x_i be j^{th} the evaluation of x_i :

$$K = \sum_j x_i^{(j)} x_i^{(j)^T} p(x_i^{(j)}|x_{i-1}, u, m, z)$$

$$u = \sum_j x_i^{(j)} p(x_i^{(j)}|x_{i-1}, u, m, z)$$

$$s = \sum_j p(x_i^{(j)}|x_{i-1}, u, m, z)$$

$$\sum_{x_i} = \frac{1}{s}K - \frac{1}{s^2}uu^T \tag{3}$$

x_i is a 3-vector in x y θ and j is search space over x y θ.

III. Implementation

A. ESDF

Our ESDF algorithm receives a set of measurements from a Scan-Matcher. These measurements are either a local incremental or global loop closure pose constraint. The local incremental pose constraint serves as our motion model. For each such measurement, the state representation is augmented using the Eqn.4, where u_{t+1} comes from laser odometry. For each global loop closure constraint, the state representation is updated according to the observation model described in Eqn.5. Pseudo-code is shown in Fig.3 and a flowchart of our ESDF implementation is shown in Fig.2.

Motion Model:

$$x_{t+1} \cong f(\mu_{x_t}, u_{t+1}) + F(x_t - \mu_{x_t}) + w_t$$

$$\mu_{x_t} = \begin{bmatrix} x_{x_{t+1}} \\ y_{x_{t+1}} \\ \theta_{x_{t+1}} \end{bmatrix} \qquad u_{t+1} = \begin{bmatrix} \delta_x \\ \delta_y \\ \delta_\theta \end{bmatrix}$$

$$\begin{bmatrix} x_{x_{t+1}} \\ y_{x_{t+1}} \\ \theta_{x_{t+1}} \end{bmatrix} = \begin{bmatrix} \delta_x \\ \delta_y \\ \delta_\theta \end{bmatrix} + \begin{bmatrix} \delta_x cos\theta_{x_t} - \delta_y sin\theta_{x_t} \\ \delta_x sin\theta_{x_t} + \delta_y cos\theta_{x_t} \\ \delta_\theta \end{bmatrix}$$

$$F = \begin{bmatrix} 1 & 0 & -\delta_x sin\theta_{x_t} - \delta_y cos\theta_{x_t} \\ 0 & 1 & \delta_x cos\theta_{x_t} + \delta_y sin\theta_{x_t} \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

```
Initialize State Variables  η, Λ, μ, Σ
while (1)
    u = get measurement from scan matcher
    i, j = poses from measurement
    if j not equal to i + 1
        Apply measurement update equations
        Do full state recovery
    else
        Propagate state from t to t + 1
        Apply state augmentation equations
        Do partial state recovery
End
```

Fig. 3. Pseudo-code for ESDF

Observation Model:

$$\mu_{x_i} = \begin{bmatrix} x_{x_i} \\ y_{x_i} \\ \theta_{x_i} \end{bmatrix}, \quad \mu_{x_j} = \begin{bmatrix} x_{x_j} \\ y_{x_j} \\ \theta_{x_j} \end{bmatrix}$$

$$\hat{z} = \begin{bmatrix} (x_{x_j} - x_{x_i})cos\theta_{y_i} + (y_{x_j} - y_{x_i})sin\theta_{x_i} \\ -(x_{x_j} - x_{x_i})sin\theta_{x_i} + (y_{x_j} - y_{x_i})cos\theta_{x_i} \\ \theta_{x_j} - \theta_{x_i} \end{bmatrix}$$

$$H = \begin{bmatrix} 0 & \dots & H1 & \dots & 0 & \dots & H2 & \dots & 0 \end{bmatrix}$$

$$H1 = \begin{bmatrix} cos\theta_{x_i} & sin\theta_{x_i} & 0 \\ -sin\theta_{x_i} & cos\theta_{x_i} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$H2 = \begin{bmatrix} -cos\theta_{x_i} & -sin\theta_{x_i} & -(x_{x_j} - x_{x_i})sin\theta_{x_i} + (y_{x_j} - y_{x_i})cos\theta_{x_i} \\ sin\theta_{x_i} & -cos\theta_{x_i} & -(x_{x_j} - x_{x_i})cos\theta_{x_i} - (y_{x_j} - y_{x_i})sin\theta_{x_i} \\ 0 & 0 & -1 \end{bmatrix} \tag{5}$$

State Recovery: We recover state by solving a linear system of equations using the Matlab backsolve. In the future, we will be implementing SLAM using the iSAM framework given its computational benefits.

B. Correlation Scan-Matcher

We have implemented a Correlative Scan-Matcher to extract rigid body transformations between sequential returns from the laser range finder to obtain laser odometry. For computational efficiency a rasterized cost table of size 256 x 256 pixels is created. Initially the raster table was built representing 8m x 8m in the world and hence each pixel represented 8/256m = 0.03125m, but later we increased it to represent 16m so that the robot could see the end of corridors.

Fig. 4. Rasterized cost table for the model scan. Bright values indicate large probabilities

Fig. 5. Current scan

Initially, we created two tables; one for the model (previous) scan and one for the current scan. The table representing the model scan consisted of log probabilities of each (x,y) position in the world with radially symmetric gaussian noise. In our first implementation, we created raster tables for both the previous scan and the current scan and then performed a matrix dot product over the two 256x256 raster tables. This was highly inefficient. The computation time for various search space is shown in fig.8. The rasterized cost table of the model (prior) scan is shown in fig.4, while the table for the current scan being registered is shown in fig.5. Fig.6 is a projection of Fig.5 for a particular search value in search space over Δx,Δy, and $\Delta\theta$. In this approach we subsampled the complete dataset and stored each rigid body constraint returned by the Scan-Matcher. This injected quantization error into the final map as can be seen in fig.18.

Later we implemented the Scan-Matcher using four nested loops. The inner most loop iterated over all the query points. The psuedo code for the second approach is shown in Fig.7. In this approach we processed each scan and added a pose to our map only when a threshold was crossed. The threshold was set to a Euclidian distance

Fig. 6. Transformed current scan over a search space

- ScanMatch(pointsA,pointsB)
  - for theta: all candidate theta value
    PointsT = Transform pointsA by theta
    For all candidate x and y
    Compute logProb for each point in PointsT translated by x and y
  Return T with maximum logProb

Fig. 7. Pseudo-code for the Scan-Matcher

of $80cms$ or angular difference of $15\deg$ between poses. The value for the previous computation was used as a prior for the next computation. This basically means that if we processed two scans and got a rigid body transformation in $x\ y\ \theta$ which did not cross the threshold, we use this value as the center of our search space for the next search. By increasing the threshold, quantization noise decreases greatly as can be seen in Fig.19. Thresholding also reduced the number of poses stored in the final map. The time required to match scans by running a fourth loop were orders of magnitude faster than the previous method, shown in Fig.9. We originally

search space	time taken
0.5m, 20 degrees	1.431s
1.0m, 20 degrees	6.232s
1.0m, 40 degrees	22.459s
2.0m, 40 degrees	40.367s

Fig. 8. Time taken to match scan using matrix dot product

search space	time taken
0.5m, 20 degrees	0.06 seconds
1.0m, 20 degrees	0.26 seconds
1.0m, 40 degrees	0.55 seconds
2.0m, 40 degrees	1.61 seconds

Fig. 9. Time taken to match scan using 4 nested loops

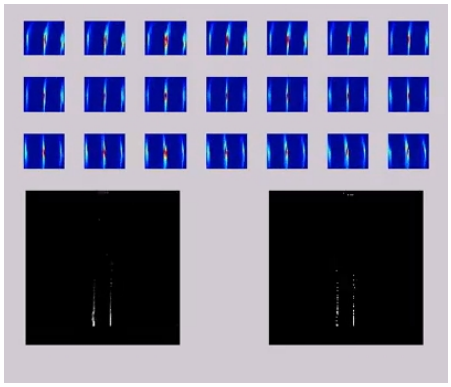

Fig. 10. Covariance estimate for less constrained environemnt

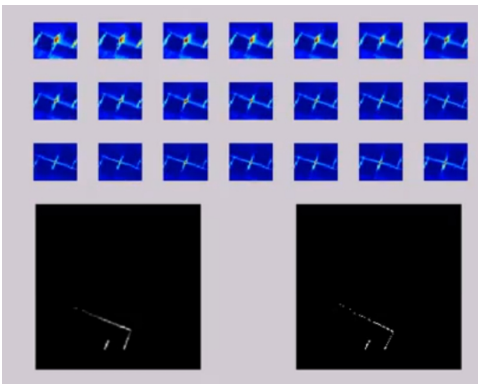

Fig. 11. Covariance estimate for highly constrained environemnt

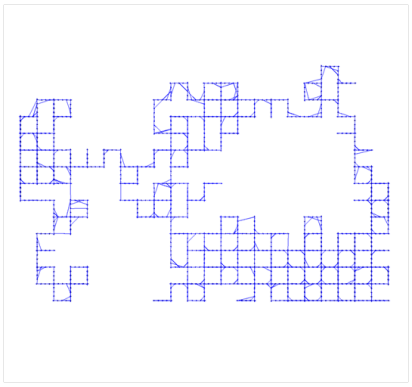

Fig. 12. Manhattan3500

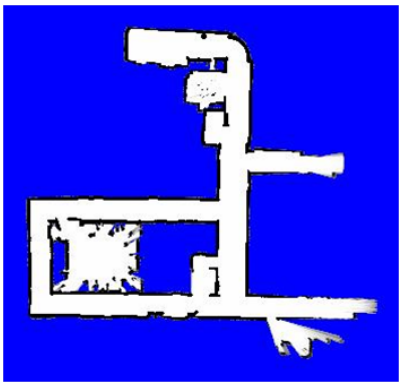

Fig. 13. CMU Newell-Simon Hall, A Level

avoided this method as we were under the impression that four nested loops in Matlab would be slow. However performing a huge matrix dot product and then summing the elements of a matrix proved less efficient.

In [2], raster tables of multiple resolutions are built. The only benefit of this approach is the computational speed. Our approach consisted of a single resolution raster table. We recovered uncertainty covariances using Eqn.3. Fig.10 and Fig.11 show Correlation (3D) Cost Function for the complete search space. The intensity plot represents the probability distribution across a search space. The figures contain 21 tiles each representing a search over θ while each pixel in a tile represents search over x and y for a given θ.

IV. Experiments

We ran our algorithms on two data sets -

Manhattan3500[3]
CMU Newell-Simon Hall, A Level[5]

Manhattan 3500 is a simulated data set of a robot driving around the streets of Manhattan. It contains 3500 pose constraints and covariances associated with each constraint. The CMU Newell-Simon Hall dataset contains odometry from an IMU and range and bearing returns for a 180 degree SICK laser.

Manhattan 3500: This dataset was used to verify the performance of the ESDF before fusing it with the correlative Scan-Matcher to complete our system. Fig.14 and Fig.15 below show the ESDF performance both with and without loop closures.

We can observe the natural sparsity of the SLAM information matrix in Fig.16. Only 0.182 percent of the elements of the information matrix are non-zero.

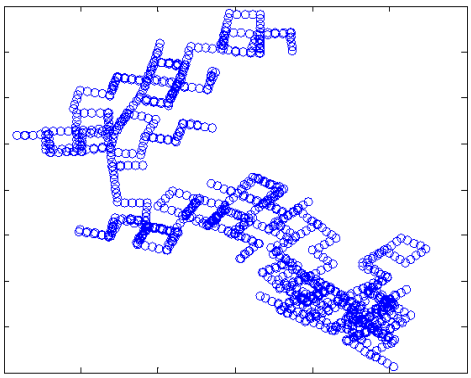

Fig. 14. Manhattan open loop

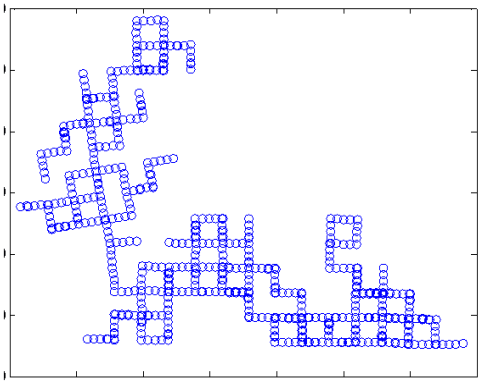

Fig. 15. Manhattan close loop

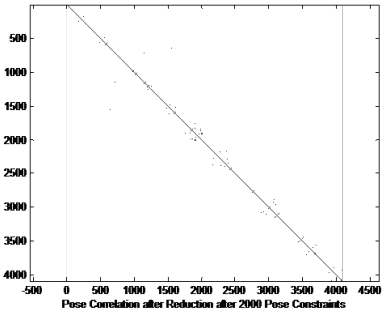

Fig. 16. Information matrix topology

State recovery currently requires us to invert the information matrix using MATLABs backslash operator. While this method is efficient in terms of matrix inversion, the algorithm is of quadratic complexity. We pay for this computationally as can be seen below in Fig.17. In order to make our system online, we will need to greatly increase the speed at which we recover the state of the robot.

CMU Newell-Simon Hall: This dataset was first used to verify the performance of the correlative Scan-Matcher in an open loop scenario. Once it was confirmed that the Scan-Matcher was returning reliable rigid body

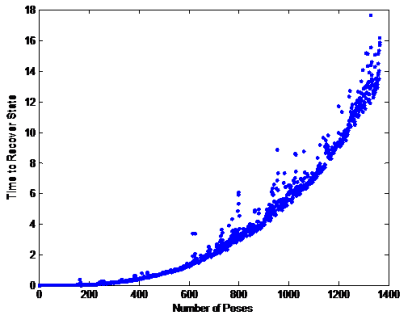

Fig. 17. Computational time required to recover state

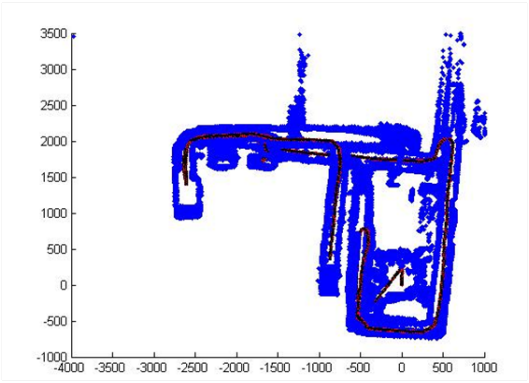

Fig. 18. CMU map recovered using matrix dot product

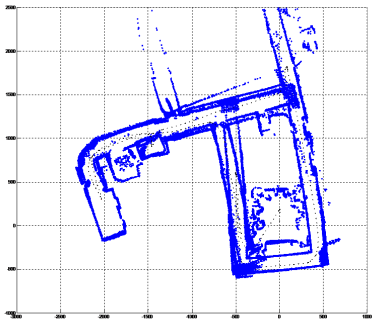

Fig. 19. CMU map recovered using 4 nested loops

constraints, we used the rigid body constraints from the Scan-Matcher in conjunction with the ESDF. Fig.19 below shows the map generated using only the pose constraints from the correlative Scan-Matcher. Fig.18 is from an older version of the Scan-Matcher. The improvement is apparent in Fig.19.

Fig.20 displays the closed loop performance of the ESDF on the pose constraints generated by the Scan-Matcher. The system runs open loop until the final thirteen pose constraints, which are loop closing events.

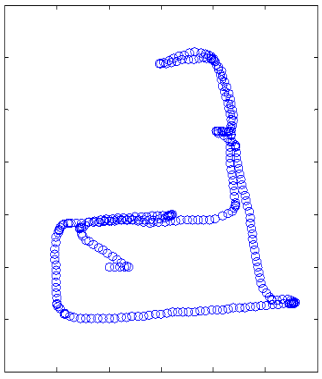

Fig. 20. CMU map recovered after running ESDF on it

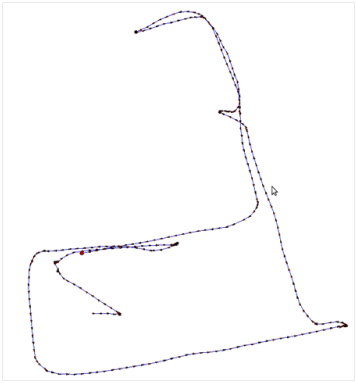

Fig. 21. CMU dataset on iSAM - openloop

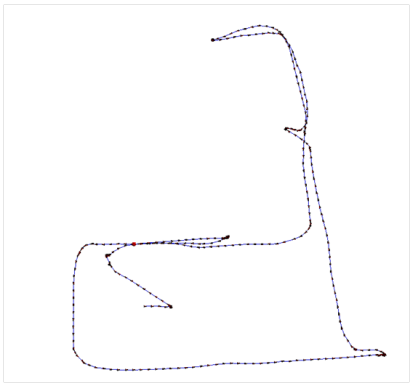

Fig. 22. CMU dataset on iSAM - closeloop

Comparison to ISAM: The laser odometry with and without loop closure was run on the iSAM framework as shown in Fig.21 and Fig.22. For our final implementation we would like to use iSAM given its computational benefits.

V. Conclusion

Michigan Autonomous Aerial Vehicles requires a SLAM implementation for an autonomous quadrotor UAV designed to fly in an office environment. In this report we have presented a solution that incorporates an Exactly Sparse Delayed-State Filter for pose-graph SLAM with constraints between poses in the map extracted from a Correlative Scan-Matcher. Most importantly, we have implemented a system that does not depend on the inherently unreliable motion model for a quadrotor and also does not rely on existence of distinct features in the environment that can be used for loop closures. The method is theoretically sound but requires work to improve computational efficiency if we are to make the system online. Some measures that will be taken include porting the implementation to a compiled coding language such as C++ or Java, as well as taking advantage of existing proven efficient frameworks such as iSAM.

References

[1] R. Eustice et al, Exactly Sparse Delayed-State Filters for View-Based SLAM

[2] Edwin Olson, "Real-Time Correlative Scan Matching"

[3] $iSAMv1.5$, http://people.csail.mit.edu/kaess/isam/

[4] M. Kaess, A. Ranganathan, and F. Dellaert, Isam:Incremental Smoothing and Mapping

[5] RADISH: The Robotics Data Set Repository,http://radish.sourceforge.net/

[6] R.Smith and P.Cheeseman. On the representation and estimation of spatial uncertainty. Intl. J. of Robotics Research, 5(4):5668, 1987.

[7] S.Thrun, W.Burgard, and D.Fox. Probabilistic Robotics. The MITpress, Cambridge, MA, 2005.

[8] P. Besl and N. McKay, A method for registration of 3-d shapes, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 14, no. 2, pp. 239256, 1992.

[9] S. Thrun, M. Diel, and D. ahnel, Scan alignment and 3d surface modeling with a helicopter platform, 2003. [Online]. Available: citeseer.ist.psu.edu/thrun03scan.html

[10] M. C. Bosse, ATLAS: a framework for large scale automated mapping and localization, Ph.D. dissertation, Massachusetts Institute of Technol- ogy, Cambridge, MA, USA, February 2004.

[11] E. Olson, Robust and efficient robotic mapping, Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, USA, June 2008.

[12] A. Censi, An icp variant using a point-to-line metric, in Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2008