Lidar-Lidar Hand-Eye Calibration via Optimization on SE(3)

Dakota Wenberg*, Aaron Ray*, Ryan Sander*
Massachusetts Institute of Technology

***Abstract*— As more autonomous systems become equipped with sensors for remote sensing, it is imperative we have methods for fast and efficient online calibration between these sensors. This is particularly important for systems with odometric drift and time-varying sensor extrinsics. In this paper, we implement a lidar-lidar calibration framework for use in the DARPA Subterranean Challenge. Using informed nonlinear on-manifold optimization to solve for an optimal relative pose in tandem with pose visualizations and both internal and external calibration scoring criteria, we develop a self-contained estimation and evaluation framework for estimating and determining the quality of lidar calibration.**

I. INTRODUCTION

Online calibration between sensors is a crucial task for autonomous systems navigating and performing tasks in the world, particularly when autonomous systems exhibit odometric drift or time-varying sensor extrinsics. In this paper, we investigate lidar-lidar calibration, an absolute orientation problem in which we calculate the rigid body pose transformation between two Velodyne lidar sensors outfitted on an autonomous robot for the DARPA Subterranean Challenge.

To solve this problem, we consider the method of "closing the pose loop" by chaining together odometric measurements between two timesteps and two coordinate systems. This problem is often referred to as the "$\mathbf{AX} = \mathbf{XB}$" problem, because applying a transform from time $t = i$ to $t = i + 1$ (pose $\mathbf{A}$) before transforming into a sensor frame (pose $\mathbf{X}$) should be the same as first transforming into the sensor frame (pose $\mathbf{X}$) and then transforming to the next timestep (pose $\mathbf{B}$).

We formulate the problem as minimizing $\left\|\mathbf{X} - \mathbf{A}^{-1}\mathbf{X}\mathbf{B}\right\|$, subject to $\mathbf{X} \in \mathbf{SE}(3)$, where $\mathbf{X}$ represents the relative pose between two lidar sensors.

II. RELATED WORK

Previous approaches to solving hand-eye calibration problems can generally be grouped into two classes of algorithms: closed-form approaches and iterative.

Closed-Form Methods

Closed-form methods aim to solve the Hand-Eye Calibration problem without the use of iterative optimization procedures. This is typically accomplished by first estimating rotation $\hat{\mathbf{R}}$, fixing this rotation estimate, and using this fixed estimate to estimate translation $\hat{\mathbf{t}}$ [8][10]. While these solutions allow for finding intuitive solutions without the need for iterative optimization, because the translation estimate is constrained by the fixed rotation estimate, any measurement errors associated with the rotation estimate will be propagated to the translation estimate as well, thereby producting sub-optimal relative transform estimates.

*All authors denoted contributed equally.

Iterative Optimization

Other approaches aim to solve the lidar-lidar calibration problem by jointly estimating the robot and sensor motion. This method can optimize the calibration between all lidar sensors simultaneously, providing potentially more robust estimates than estimating pairwise transformations. This problem is formulated using Pose Graph Optimization (PGO), as in [2]. The resulting problem is solved with a local nonlinear solver such as Gauss-Newton or Levenberg-Marquadt. An initial guess is provided by a RANSAC-type consensus algorithm. We implement a version of this iterative optimization, limited to pairwise estimation and decoupled sensor and robot motion.

III. METHODOLOGY

To estimate this lidar-lidar calibration, we leverage a nonlinear optimization approach that takes into account both odometry data and the relative noise of each sample. Since we solve for an optimal pose, we optimize over the Special Euclidean, specifically, **SE**(3) group.

Relative Pose Estimation

We assume access to motion estimates in the two lidar frames at each timestep ($\mathbf{A}_i$, $\mathbf{B}_i$). In practice, these are calculated using the Iterative Closest Point (ICP) algorithm [11] and are already present in our dataset. As we expect that $\mathbf{T}_A^B\mathbf{A}_i\mathbf{T}_B^A\mathbf{B}_i^{-1} = \mathbf{I}$, we find an estimate for $\mathbf{T}_A^B$ by minimizing the weighted chordal distance $\left\|\mathbf{T}_A^B - \mathbf{B}_i\mathbf{T}_A^B\mathbf{A}_i^{-1}\right\|_{\mathbf{\Omega}}^2$ where $\|\mathbf{M}\|_{\mathbf{\Omega}}^2 = \text{tr}(\mathbf{M}\mathbf{\Omega}\mathbf{M}^T)$. We model rotation noise in motion measurements $\mathbf{A}_i, \mathbf{B}_i$ as a Langevin distribution with concentration parameter ω[1], and translation noise as a gaussian distribution with variance $1/\rho$ in each dimension. We introduce an additional *rejection factor* $r_i \in \{0, 1\}$, which determines if a sample should be rejected based on a threshold value of its ICP covariance estimate. The resulting optimization problem is thus given by:

$$\hat{\mathbf{T}}_A^B = \arg\min_{\mathbf{T}_A^B \in \mathbf{SE}(3)} \frac{1}{N}\sum_{i=1}^{N} r_i \left\|\mathbf{B}_i\mathbf{T}_A^B\mathbf{A}_i^{-1} - \mathbf{T}_A^B\right\|_{\mathbf{\Omega}_i}^2 \quad (1)$$

$$\text{Where } \mathbf{\Omega}_i = \begin{bmatrix}\omega_i\mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3^T & \rho_i\end{bmatrix}, \omega_i, \rho_i \in \mathbb{R}^+, r_i \in \{0, 1\} \quad (2)$$

IV. EXPERIMENTAL EVALUATION

We apply our lidar-lidar registration framework to the DARPA Subterranean Challenge, in which autonomous systems explore cavernous subterranean paths using lidar depth sensors. We leverage robot odometry data in tandem with

[1]This quantity is estimated by the inverse covariance of the angles between sequential poses.

lidar point cloud data collected from these sensors to compute the relative rigid body pose transformation between our two lidar sensors. In addition to implementing a calibration system that can be used to estimate optimal relative poses between lidar sensors, we also implement a novel evaluation framework. We implement the nonlinear manifold optimization in Equation (1) with `pymanopt` [9], which provides easy access to automatic differentiation with Python's `autograd` [6] and `numpy` [3] libraries.

Internal Model Evaluation: Weighted RMSE

To quantitatively assess the performance of our pose estimate following nonlinear optimization, we compute the weighted root mean square error (RMSE) of our pose estimate, given by:

$$T_{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left\|\mathbf{B}_i\hat{\mathbf{T}}_A^B\mathbf{A}_i^{-1} - \hat{\mathbf{T}}_A^B\right\|_{\mathbf{\Omega}_i}^2}$$

Note that here, we do not use a rejection factor $r_i \in \{0, 1\}$.

K-Fold Cross-Validation To Test Overfitting

To test the degree of overfitting our model has to a trajectory, as well as to test the robustness of our lidar-lidar calibration on unseen data, we fit the trajectory using supervised cross-validation. We ran cross-validation experiments with 5, 10, and 20 folds. For each fold, we employ a "leave-one-out" approach, in which we estimate a relative transformation between lidar sensors using all but the k$^{\text{th}}$ fold for training data, and then evaluate the performance of this relative pose estimate using only the k$^{\text{th}}$ fold for testing data. Table I illustrates the RMSE and cross-validation results.

Point Cloud Fitness Scoring and Perturbation Testing

In addition to the RMSE-based error metrics for internal calibration consistency, we validate the calibration by relating it directly to point cloud quality. We integrate each Lidar scan into a single point cloud with Octomap [4], an octree-based occupancy map[2]. An error metric is calculated between two point clouds by finding the closest point in the second cloud for each point in the first cloud, and averaging the distances[3].

This error metric is simple to set up and is intuitive, but it is naive in that it assumes each point in one point cloud should have a direct match in the other point cloud. A simple max-distance heuristic partially addresses this concern, but it is not immediately clear if the metric is useful to compare calibration quality. To test the sensitivity of this metric, we compared the known-good hand-calibrated lidar alignment with varying amounts of rotation noise in the transform. Figure 1 illustrates this relationship. The unperturbed good transformation exhibits noticeably lower error than any perturbed transformation. This demonstrates that while the absolute error scale may be inaccurate due to distortions from the max-distance heuristic, this error metric can still be used to detect deviations from optimal calibration and compare competing calibration methods. We also note that Figure 1 shows that our optimized calibration appears to be worse than the hand-calibrated version. We attribute this to two possible causes. First, treating the motion estimate for each lidar separately (instead of both caused by the same robot motion) may result in less robustness to noise. Second, the robot's motion is mostly planar, so it may have not excited enough degrees of freedom to get good estimates for all calibration parameters - indeed, we see our optimized solution is quite close to the hand-tuned version in all dimensions except vertical translation.

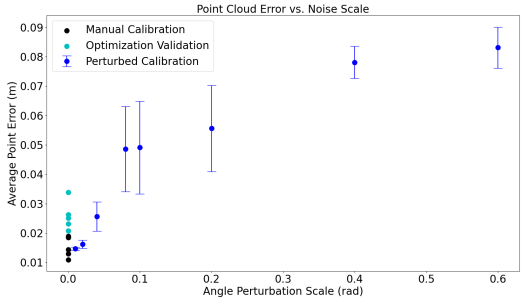

Fig. 1. Point cloud alignment error for hand-tuned, optimized, and perturbed sensor calibrations. Relative error is more informative than absolute magnitude, due to outlier-rejection effects.

TABLE I
AVERAGE RMSE CROSS-VALIDATION RESULTS FOR 5, 10, 20 FOLDS

Estimate	RMSE $\mathbf{R}$, Weighted	RMSE $\mathbf{t}$, Weighted
Initial, 5 Folds	5.4087e-04 rad.	1.5318e-03 m.
Final, 5 Folds	5.4081e-04 rad.	1.5204e-03 m.
Initial, 10 Folds	5.3344e-04 rad.	1.5181e-03 m.
Final, 10 Folds	5.3340e-04 rad.	1.5070e-03 m.
Initial, 20 Folds	5.0450e-04 rad.	1.4622e-03 m.
Final, 20 Folds	5.0448e-04 rad.	1.4503e-03 m.

V. RECOMMENDATIONS AND CONCLUSION

We have shown that while in principle the Hand-Eye Calibration problem can be solved with a simple manifold optimization formulation, finding the optimal calibration may require a more sophisticated approach. In particular, the solution presented in [2] appears to be a viable option. We have also shown the value in a secondary evaluation method based on point cloud alignment to validate calibration quality. We recommend refining the point cloud error metric to use point cloud feature alignment instead of closest point. This will allow for straightforward analysis of more sophisticated calibration methods. To address the issue of underexcited degrees of freedom, we would want to try using ground plane estimation to directly solve for 3 degrees of freedom

[2]We have noticed that for the short segments of data we have worked with, voxel downsampling in a normal point cloud library like PCL works just as well, as there are few outlier scans

[3]`https://pointclouds.org/documentation/classpcl_1_1registration_1_1_transformation_validation_euclidean.html`

(roll, pitch, and z), and solving for the remaining three degrees of freedom with nonlinear optimization. We expect this method to perform well when the robot can be initialized in a reasonably flat area.

We have released our lidar-lidar calibration framework as an open source project on Github[4].

ACKNOWLEDGEMENTS

We would like to thank Professor Luca Carlone and the VNAV staff for support on this project. Additionally, we thank Benjamin Morrell and Fadhil Ginting for providing data and introducing us to the DARPA Sub-T challenge. Finally, we would like to thank Mike Schoder and Phil Cotter for collaborating and coordinating with us on this project.

REFERENCES

[1] F. Dellaert, "Factor graphs and gtsam: A hands-on introduction," Georgia Institute of Technology, Tech. Rep., 2012.

[2] F. Furrer, M. Fehr, T. Novkovic, H. Sommer, I. Gilitschenski, and R. Siegwart, "Evaluation of combined time-offset estimation and hand-eye calibration on robotic datasets," in *Field and Service Robotics*, Springer, 2018, pp. 145–159.

[3] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del R'ıo, M. Wiebe, P. Peterson, P. G'erard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020. DOI: 10.1038/s41586-020-2649-2. [Online]. Available: https://doi.org/10.1038/s41586-020-2649-2.

[4] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous robots*, vol. 34, no. 3, pp. 189–206, 2013.

[5] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007. DOI: 10.1109/MCSE.2007.55.

[6] D. Maclaurin, D. Duvenaud, and R. P. Adams, "Autograd: Effortless gradients in numpy."

[7] MATLAB, *version 7.10.0 (R2010a)*. Natick, Massachusetts: The MathWorks Inc., 2010.

[8] Y. C. Shiu and S. Ahmad, "Calibration of wrist-mounted robotic sensors by solving homogeneous transform equations of the form ax= xb.,"

[9] J. Townsend, N. Koep, and S. Weichwald, "Pymanopt: A python toolbox for optimization on manifolds using automatic differentiation," *Journal of Machine Learning Research*, vol. 17, no. 137, pp. 1–5, 2016. [Online]. Available: http://jmlr.org/papers/v17/16-177.html.

[10] R. Y. Tsai, R. K. Lenz, *et al.*, "A new technique for fully autonomous and efficient 3 d robotics hand/eye calibration," *IEEE Transactions on robotics and automation*, vol. 5, no. 3, pp. 345–358, 1989.

[11] Z. Zhang, "Iterative closest point (icp)," in *Computer Vision: A Reference Guide*, K. Ikeuchi, Ed. Boston, MA: Springer US, 2014, pp. 433–434, ISBN: 978-0-387-31439-6. DOI: 10.1007/978-0-387-31439-6_179. [Online]. Available: https://doi.org/10.1007/978-0-387-31439-6_179.

[4]Our code can be found at https://github.com/rmsander/lidar-lidar-cal

APPENDIX

A1. Computing Rejection Factors

The rejection factor $r_i \in \{0, 1\}$ is computed by determining if the maximum diagonal element of the 6x6 ICP covariance matrix for the i$^{\text{th}}$ sample $\Sigma_{\text{ICP},i}$ is above a given threshold $\tau \in \mathbb{R}^+$. Mathematically:

$$r_i(\Sigma_{\text{ICP},i}) = \begin{cases} 0, & \max(\text{Diag}(\Sigma_{\text{ICP},i})) > \tau \\ 1, & \text{o.w.} \end{cases}$$

In other words, if any diagonal element of the ICP covariance matrix exceeds τ, we reject the sample based on our belief that the sample is too noisy. For our experiments, this threshold $\tau = 100$.

A2. Robot and Sensor Setup

The following diagrams describe our robot setup and the lidar sensor placement on the robots used for the DARPA SubT challenge.

Fig. 2. Photograph of the Husky robots used to collect data for the DARPA SubT Challenge. Note the three lidars that each of these "Husky" robots possesses. Notice that the "rear" lidar has more occlusion from other pieces of robot hardware than the "main" and "front" lidar sensors.

A. A3. Methodology and Results

The following diagrams illustrate the Hand-Eye Calibration problem we aim to solve, our specific methodologies used for solving it, and our results from our calibration system.

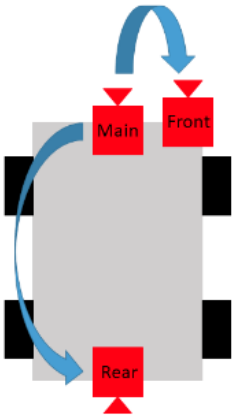

Fig. 3. The 3-lidar configuration for a Husky robot, from a bird's-eye view. The arrows indicate some of the pairwise relative transforms we wish to compute through optimal pose estimation.

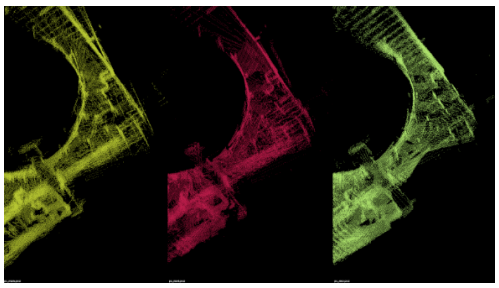

Fig. 4. Sample point clouds produced by scans from the 3 different lidar sensors. Our goal with this project was to calibrate the relative transform between the lidar sensors in order to perform point cloud alignment.

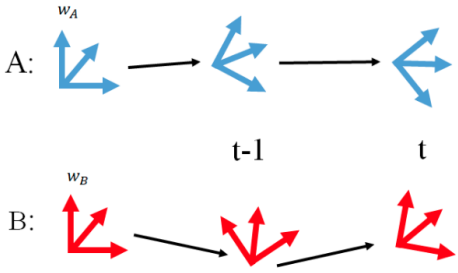

Fig. 5. Our Hand-Eye calibration problem: Given frames of reference, in this case, lidar sensors A and B, and their relative pose transformations over time from odometry data, we can compute the relative transform between these two lidar sensors. This optimization is carried out in 1.

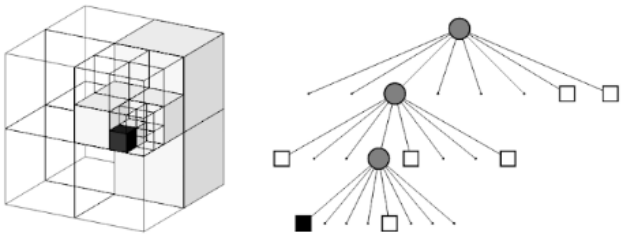

Fig. 6. Conceptual idea of how octrees work for efficiently representing voxel-based representations of scenes and objects. Note that the branching factor of the graph on the right is 8, since each voxel unit can be sub-divided into 8 octants in 3D space, provided the voxel unit is not a leaf node. Source: [4].

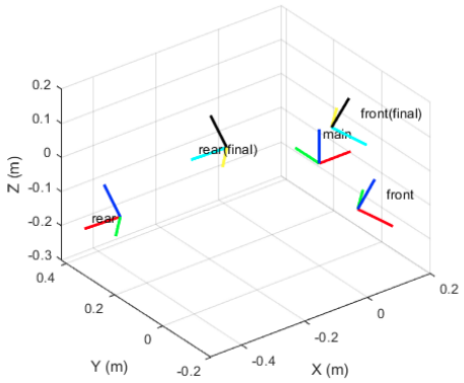

Fig. 7. A 3D visualization of our weighted pose estimates. Note that the rotation between the initial and final estimates is nearly the same for each sensor, but the translation differs significantly. We believe this to be due to the largely planar odometric relative motion of the robots as they explore. Graph generated using `MATLAB` [7].

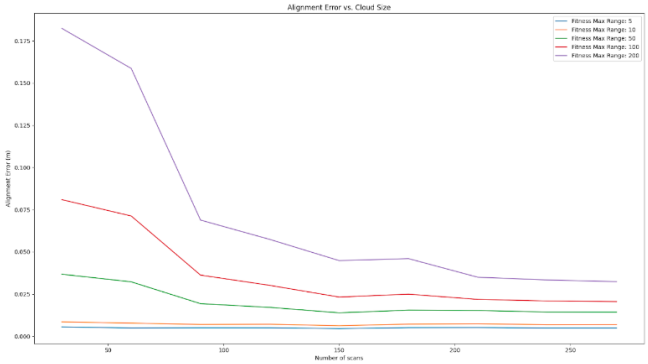

Fig. 8. Plot illustrating point cloud alignment error for different maximum ranges of fitness scores. We observed that the higher the fitness score, the lower the alignment error. Additionally, the more scans we ran, the lower the alignment error. Plot generated using `matplotlib` [5].

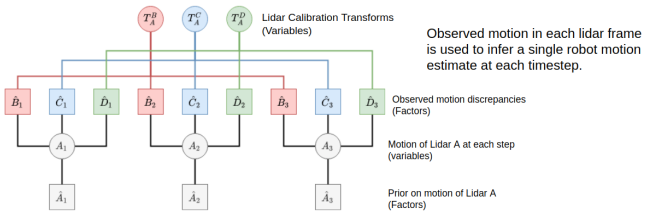

Fig. 9. An extension of our work, and a future avenue of it, is based on using Pose Graph Optimization (PGO) to calibrate between several lidar sensors at once. This is an advantageous future work step for this project because it (i) Enables for solving for multiple relative transforms at once, (ii) Since we have more measurements, we are able to produce an improved result for the joint relative transform estimates compared to the pairwise, decoupled relative transform estimates. A PGO approach such as this can be carried out using the `GTSAM` library [1].