Continuous Trajectory Estimation for 3D SLAM from Actuated Lidar*

Hatem Alismail, L. Douglas Baker and Brett Browning

***Abstract*— We extend the Iterative Closest Point (ICP) algorithm to obtain a method for continuous-time trajectory estimation (CICP) suitable for SLAM from actuated lidar. Traditional solutions to SLAM from actuated lidar rely heavily on the accuracy of an auxiliary pose sensor to form rigid frames. These frames are then used with ICP to obtain accurate pose estimates. However, since lidar records a single range sample at time, any error in inter-sample sensor motion must be accounted for. This is not possible if the frame is treated as a rigid point cloud. In this work, instead of ICP we estimate a continuous-time trajectory that takes into account inter-sample pose errors. The trajectory is represented as a linear combination of basis functions and formulated as a solution to a (sparse) linear system without restrictive assumptions on sensor motion. We evaluate the algorithm on synthetic and real data and show improved accuracy in open-loop SLAM in comparison to state-of-the-art rigid registration methods.**

I. INTRODUCTION

Point cloud registration methods play an important role in front-end SLAM from lidar. In particular, the Iterative Corresponding/Closest Point (ICP) algorithm [5] and its variants [22] have been used heavily for sensor pose estimation prior to back-end optimization (*c.f*. [9]). One particularly common sensor for 3D SLAM is "actuated lidar", where a 2D scanning lidar is actuated to sweep a volume in space. Actuated lidar remains popular due to its lower cost and flexibility in comparison to other 3D sensors.

A major limitation of actuated lidar is the serial acquisition of 3D points. Since the robot is typically moving while the sensor is being actuated, the resulting point clouds will be distorted. This problem is commonly solved by coupling the lidar with a pose sensor, such as an Inertial Measurement Unit (IMU), or odometry, *etc*. Given the pose estimates, we may *undistort* a segment of the lidar data and accumulate it into a rigid *frame*. This frame can then be registered to one, or more, prior frames allowing us to estimate the pose of the sensor and build a map of the environment.

In such methods, drift in pose estimates, inadequate sampling density, or dropouts will be integrated into the rigid frames. Such errors cannot be corrected via ICP and can significantly affect the accuracy of the system (*c.f*. Fig.(1)).

In this work, we propose a new registration technique that explicitly accounts for sensor motion. In contrast to rigid registration, we develop an algorithm that estimates a *continuous* 6DOF trajectory of the sensor. The trajectory spans the duration of frame acquisition and can potentially assign a distinct pose per measurement. The algorithm is shown to improve the accuracy of open-loop SLAM in comparison to rigid registration. Our algorithm extends the familiar ICP framework alternating between correspondences and pose estimation. While ICP estimates a single rigid pose, our algorithm estimates a continuous-time trajectory, or a Continuous-ICP (CICP).

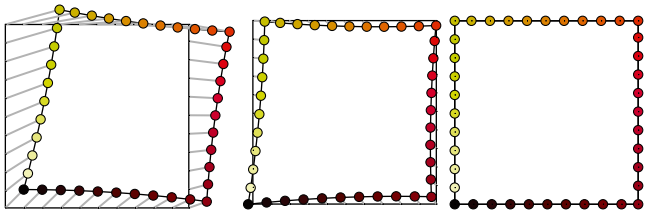

Fig. 1: From left, scanned points from a moving sensor are indicated by ○ and colorized by acquisition time in a square environment (solid). Gray lines indicate correspondences. ICP (center) fails to take into account the continuous sensor motion while CICP (right) correctly aligns the point sets.

In CICP, the trajectory is represented with a linear combination of basis functions and solved efficiently as a linear system. The solution does not use the small-angle approximation to linearize rotations, which means we are not limited to small motions. Furthermore, in addition to efficiency, it is easy to implement robust solvers via regularization methods, which is desirable in registration problems in presence of noise and missing correspondences. Next, we present the formulation of the problem.

II. PRELIMINARIES

Given two sets of 3D points in correspondence, a set of "stationary" points $\mathcal{S} = \left\{\mathbf{s}_i \in \mathbb{R}^3\right\}_{i=1}^{m}$ and a set of "moving" points $\mathcal{M} = \left\{\mathbf{m}(t_i) \in \mathbb{R}^3\right\}_{i=1}^{m}$ where $\mathbf{s}_i \leftrightarrow \mathbf{m}(t_i)$, we seek to estimate a trajectory $\mathbf{T} : \mathbb{R} \rightarrow SE(3)$, such that

$$\mathbf{s}_i = \mathbf{T}(t_i)\mathbf{m}(t_i), \tag{1}$$

where $\mathbf{T}(t) \in SE(3)$ is a rigid body transform composed of a rotation $\mathbf{R}(t) \in SO(3)$ and a translation $\mathbf{t}(t) \in \mathbb{R}^3$.

Clearly, the problem in Eq.(1) is underconstrained and cannot be solved directly. Instead, we propose to model the trajectory as a linear combination of n "control" poses $\tilde{\mathbf{T}}_j$ at distinct time samples $\tilde{\mathbf{t}} = \left\{\tilde{t}_j\right\}_{j=1}^{n}$. Let $\mathbf{v}_j$ be a vectorial representation of each of the control pose parameters, our trajectory $\mathcal{T}$ at an arbitrary time $\tilde{t}_1 \leq t \leq \tilde{t}_n$ can be written as a linear combination of a set basis functions $\phi_j(\cdot)$ as

$$\mathcal{T}(t; \tilde{\mathbf{t}}, \{\mathbf{v}_j\}) = \sum_{j=1}^{n} \phi_j(t; \tilde{\mathbf{t}})\mathbf{v}_j. \tag{2}$$

*This Project Agreement Holder (PAH) effort was sponsored by the U.S. Government under Other Transaction number W15QKN-08-9-0001 between the Robotics Technology Consortium, Inc, and the Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the U.S. Government.

Alismail (halismai@cs.cmu.edu), Baker (ldbapp@cs.cmu.edu) and Browning (brettb@cs.cmu.edu) are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213 USA.

Given this representation, we may obtain the pose of the sensor during any time $t \in \left[\tilde{t}_1, \tilde{t}_n\right]$ by evaluating the function using the estimated coefficients/poses $\{\mathbf{v}_j\}$.

A. Linear Solution to the Rigid Pose Estimation Problem

Consider two point sets $\left\{\mathbf{y}_i \in \mathbb{R}^3\right\}_{i=1}^m$ and $\left\{\mathbf{x}_i \in \mathbb{R}^3\right\}_{i=1}^m$ in correspondence ($\mathbf{y}_i \leftrightarrow \mathbf{x}_i$) and related via a rigid body transform, we seek to estimate $\mathbf{R}$ and $\mathbf{t}$ such that

$$\mathbf{y}_i = \mathbf{R}\mathbf{x}_i + \mathbf{t}. \tag{3}$$

The problem is well-studied with various closed form solutions in the literature [12]. A linear solution can be obtained by representing the rotation using the Gibbs vector [10], or Cayley-Rodrigues. The Gibbs vector $\mathbf{g}$ encodes the rotation in terms of an Euler axis and an angle θ as $\mathbf{g} = \hat{\mathbf{g}} \tan(\theta/2)$, where $\hat{\mathbf{g}}$ is a unit norm Euler axis. In order to convert a Gibbs vector to a rotation matrix, we may use the Cayley transform given by

$$\mathbf{R} = \left(\mathbf{I}_3 + \mathbf{G}\right)^{-1}\left(\mathbf{I}_3 - \mathbf{G}\right), \tag{4}$$

where $\mathbf{I}_3$ is the 3×3 identity and $\mathbf{G} = [\mathbf{g}]_\times$ is a skew-symmetric matrix. Using this, we can can write Eq.(3) as

$$\mathbf{y}_i = \left(\mathbf{I}_3 + \mathbf{G}\right)^{-1}\left(\mathbf{I}_3 - \mathbf{G}\right)\mathbf{x}_i + \mathbf{t}. \tag{5}$$

Multiplying Eq.(5) by $(\mathbf{I}_3 + \mathbf{G})$, and rearranging we obtain

$$\mathbf{d}_i = [\mathbf{a}_i]_\times \mathbf{g} + \check{\mathbf{t}}, \tag{6}$$

where $\mathbf{d}_i = \mathbf{y}_i - \mathbf{x}_i$, $\mathbf{a}_i = \mathbf{y}_i + \mathbf{x}_i$, and $\check{\mathbf{t}} = \left(\mathbf{I} + \mathbf{G}\right)^{-1}\mathbf{t}$. Here, we used the identity $\mathbf{a} \times \mathbf{b} = [\mathbf{a}]_\times \mathbf{b} = -[\mathbf{b}]_\times \mathbf{a}$. Eq.(6) is linear and can be solved by standard linear methods.

III. Continuous Trajectory Estimation

In our original problem, Eq.(1), we seek to estimate a continuous trajectory per-point such that the Euclidean distance between corresponding points is minimized. We can formulate this problem as a linear least squares optimization as follows. First, we choose a number of "control poses" represented as vectors $\mathbf{v}_j \in \mathbb{R}^6$, where the rotation is represented with the Gibbs vector Eq.(4). The trajectory $\mathcal{T} : \mathbb{R} \rightarrow SE(3)$, at time t, can then be written as a linear combination of basis functions (Eq.(2)). Given this representation, we obtain the following linear system of equations $\mathbf{b}(t) = \mathbf{s}_t - \mathbf{m}(t) = \mathbf{\Lambda}(t;\tilde{\mathbf{t}})\boldsymbol{\theta}$, where $\boldsymbol{\theta} = \left(\mathbf{v}_1^\top, \ldots, \mathbf{v}_n^\top\right)^\top$ is a $6n$ vectorial representation of the n control poses, and $\mathbf{\Lambda}$ is a $3 \times 6n$ matrix of the linear constraints in Eq.(6) weighted by the basis functions, *i.e.*

$$\mathbf{\Lambda}(t;\tilde{\mathbf{t}}) = \begin{pmatrix}[\mathbf{s}_t + \mathbf{m}(t)]_\times & \mathbf{I}_3\end{pmatrix}\tilde{\mathbf{\Phi}}(t;\tilde{\mathbf{t}}), \tag{7}$$

where $\tilde{\mathbf{\Phi}}(t;\tilde{\mathbf{t}}) = \left(\phi(t;\tilde{t}_1)\mathbf{I}_6, \ldots, \phi(t;\tilde{t}_n)\mathbf{I}_6\right)$. Given sufficient correspondences, we obtain the following linear system of equations

$$\begin{pmatrix}\mathbf{b}(t_1)\\ \vdots \\ \mathbf{b}(t_m)\end{pmatrix} = \begin{pmatrix}\mathbf{\Lambda}(t_1;\tilde{\mathbf{t}})\\ \vdots \\ \mathbf{\Lambda}(t_m;\tilde{\mathbf{t}})\end{pmatrix}\begin{pmatrix}\mathbf{v}_1^\top\\ \vdots \\ \mathbf{v}_n^\top\end{pmatrix} \tag{8}$$

$$\mathbf{b} = \mathbf{A}\boldsymbol{\theta}. \tag{9}$$

The optimal least squares solution can be obtained using standard approaches, and can be done efficiently using elementary matrix operations.

IV. Continuous Iterative Corresponding Points

Our continuous trajectory formulation can be easily integrated into the familiar ICP framework. Hence, we may use previous research and best-practices for ICP-like algorithms [19]. Point correspondences are estimated based on the closest distance using approximate nearest neighbor search with KD-trees[1]. To improve the efficiency of the system, we filter correspondences to maintain one-to-one unique matches, with ties broken by distance. We also remove correspondence with a distance greater than a threshold.

Convergence is determined if the any of the following conditions is true (i) change in the estimated pose is smaller than a threshold 10^{-6}, (ii) change in closest points error is smaller than a threshold 10^{-6}, or (iii) a predetermined maximum number of iterations is reached 100. Thresholds are determined experimentally and held fixed. The framework is also used for the rigid registration method we compare against.

B-spline basis functions [11] are used to represent the trajectory, and implemented using Cox-de Boor recurrence formulae. The jth B-spline basis function of degree d (order $o = d + 1$), over a non-decreasing sequence of real numbers (knot vector) $\mathbf{u} = (u_1, \ldots, u_n)$ is defined by the recurrence

$$\beta_j^0(u) = \begin{cases}1 & \text{if } u_j \leq u < u_{j+1};\\ 0 & \text{otherwise.}\end{cases} \tag{10}$$

$$\beta_j^d(u) = \frac{u - u_j}{u_{j+d} - u_j}\beta_j^{d-1}(u) + \frac{u_{j+o} - u}{u_{j+o} - u_{j+1}}\beta_{j+1}^{d-1}(u) \tag{11}$$

In our implementation, the basis functions are placed uniformly. The choice of B-splines is motivated by their flexibility and approximation power [25]. More importantly, B-spline bases are compactly supported; each function has a local effect on the trajectory and hence errors do not propagate globally. Compact support yields a sparse system, allowing increased efficiency when solving Eq.(9).

V. Experiments and Results

A. Simulation

We evaluate the performance of our algorithm using data collected from a scanner [24] that are synthetically deformed using a generated non-linear trajectory to simulate a scanning-while-moving scenario (see Fig.(4)). The trajectory is constructed by generating $n = 6$ ground truth poses and interpolating in between such that each point is assigned a distinct pose. This producers distorted point clouds similar to what we obtain from a moving sensor incrementally scanning the world. Experiments are repeated 100 times with random trajectories and noise levels per trial. For every trial, we report the root mean squared error (RMSE) of the translation and rotation of the estimated trajectory. For translation, this is computed as $\epsilon_{\mathbf{t}} = \left(n^{-1}\sum_i^n \|\mathbf{t}_i - \hat{\mathbf{t}}_i\|^2\right)^{1/2}$, where $\hat{\mathbf{t}}_i$ is the estimated translation for the i^{th} trajectory pose. Similarly, rotation RMSE is computed using a vector of Euler angles.

[1]We use nanoflann `http://code.google.com/p/nanoflann/`, a fine-tuned implementation of FLANN [18] for 3D points.

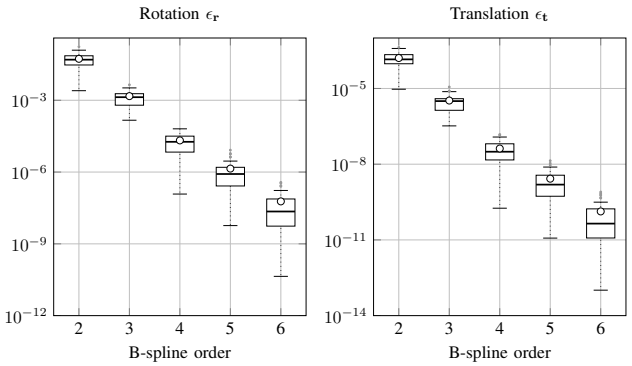

Fig. 2: Basis functions vs. accuracy (log scale).

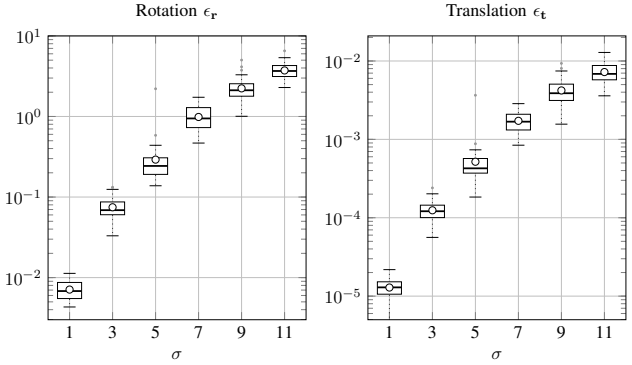

Fig. 3: Simulation results with noise and missing correspondences (log scale). For moderate noise, $\sigma < 7$, median error is less than $1/4^{\circ}$ and $1/2$mm.

B. Effect of basis functions order

The choice of the B-spline basis function order is an important parameter to the algorithm. Ideally, we desire a balance between computational requirements and the ability to approximate the true trajectory. Fig. 2 shows the results of B-spline basis of different orders. The trajectory is generated by selecting six random poses and interpolated in between with a cubic spline. Since the data is noise-free and correspondences are known, we expect a close to perfect result[2]. This is true when the order of the function is four or more. By increasing the order of the bases, we are able to attain better approximation, within numerical limits.

C. Performance under noise

Gaussian noise is added to both sets with varying std. deviation. To simulate missing correspondences, we remove 20% of the points randomly at each iteration. The experiments are performed for different types of trajectories generated at random (with linear interpolation), including pure translation, pure rotation and a combination of both. The experiments have been performed with quadratic B-spline functions (order 3). Results are summarized in Fig.(3). We can see that the algorithm performs well under moderate levels of noise with performance degrading gracefully.

[2]Perfect results are not possible since the trajectory is interpolated with a cubic-spline, which is not the same as a cubic B-spline.

D. Regularization & a robust solution

Since our continuous trajectory estimation is linear, it is easy to implement a robust variant. Robustness is desired in the context of registration due to the missing correspondences, noise and outliers. One possibility is solving the system with L_1-regularization. That is we seek $\boldsymbol{\theta}$ that minimizes the regularized error $\|\mathbf{A}\boldsymbol{\theta} - \mathbf{b}\|_2^2 + \lambda\|\boldsymbol{\theta}\|_1$, where $\lambda \in \mathbb{R}^+$, $\|\cdot\|_p$ is the p^{th} norm, $\|\mathbf{a}\|_p = \sum_i |a_i|^{1/p}$. In contrast to least squares, Eq.(9), no closed-form solution exists for L_1-regularization. However, there exist various methods that can attain a solution robustly and efficiently.Here, we use an interior-point method by Koh *et al.* [16], which is efficient and suitable for large-scale problems.

Fig.(5) shows the results of our algorithm with and without regularization when outliers are injected into the data. Similar to the previous experiments, Gaussian noise is added and 20% of the points removed at random to simulate missing correspondences. We replace 20% of the data with gross outliers and repeat the process 100 times with a random trajectory per trial. We can see L_1 regularization improves the accuracy. However, at higher levels of noise, the error becomes large, especially for rotation. Further improvements on the L_1 solution are possible by fine-tuning the regularization parameter λ. However, the improvement is not significant and does not aid our parameter selection for real data, as we do not expect such high levels of noise and outliers.

E. Real data

We evaluate the algorithm on a real-time 3D SLAM system as a replacement to rigid registration. The sensor used is shown in Fig.(6) and composed of a 270° FOV, 40 Hz Hokuyo scanner mounted on a spinning motor. The angular resolution was 0.25°, producing 1081 measurements per scan. The motor rotates about an axis orthogonal to the scanning mirror producing a *spinning* motion at 0.25 Hz.

Every half-scan rotation of the motor (2 s worth of data) produces a "frame" composed of $\approx$ 86K points. These frames are collected while the robot is moving and a pose source is required to account for the robot motion prior to registration. For this, we employ stereo visual odometry (VO) using P3P and two-frame refinement [2]. The stereo camera is calibrated wrt. the lidar [1] in an offline step. VO runs at 10 Hz providing local pose estimates to build the undistorted frames. For every frame we have ≈ 20 VO pose samples.

Our SLAM framework is local. We keep a rolling buffer of the three most recent undistorted lidar frames. Our baseline comparison uses GICP [20] to rigidly register the undistorted frames. For CICP, we use six control poses per frame distributed uniformly over acquisition time. The 2^{nd} and 3^{rd} frames in the rolling buffer are registered to the first (stationary) frame. We solve the system with L_1 regularization with $\lambda = 10^{-4}$ and cubic B-spline basis.

We have evaluated our algorithm on various data sequences from indoor and subterranean environments. Here we show results from an industrial building. Figure 7 shows a corridor scene registered using GICP (left) and our CICP method (right). We can see visible errors with GICP results,

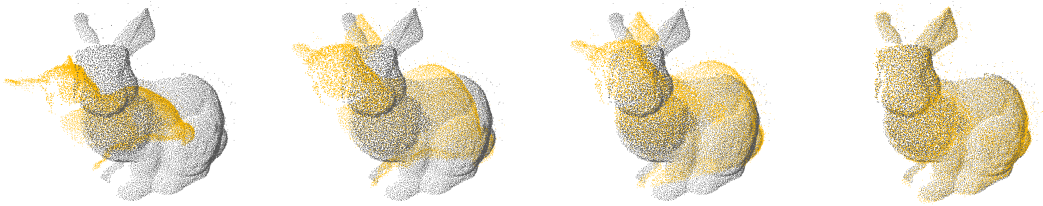

Fig. 4: A few iterations of the algorithm starting from initialization (far left), and results after iterations 1, 10 and 20 (final). Gray points are stationary. Please note the large distortion of the moving points (yellow) at the first iteration.

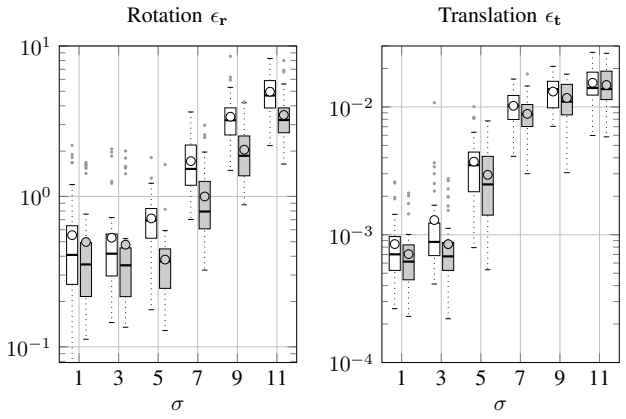

Fig. 5: Least squares performance (white) vs. L_1 regularization with 20% of outliers in addition to Gaussian noise and missing correspondences. Error is shown in log scale.

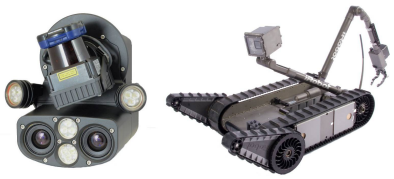

Fig. 6: Sensor assembly and robot used in this work consisting of a spinning Hokuyo and a stereo camera from Carnegie Robotics LLC mounted on an iRobot Packbot.

as rigid registration is not aware of inter-sample pose errors. In Fig.(8) is another example from the building and Fig. 9 shows a larger area from data collected in our test facility. Overall, while we do not have ground truth available, it is clear that CICP offers greater open loop accuracy.

VI. Related Work

The literature on registration methods is vast and has been active for the past two decades. We refer the reader to a recent survey by Tam *et al*. [22] for a more extensive review. Robust L_1-regularization has various applications in sparse coding and statistics. Recent work by Bach et. al [4] provides a comprehensive treatment. Recently, L_p-regularization has been proposed for rigid ICP by Bouaziz et. al [8] in which they show increased robustness.

Continuous-time estimation has been gaining popularity in Robotics and Computer Vision. Furgale et. al [13] proposed a continuous-time batch estimation framework for camera-IMU calibration. We share the use of B-spline functions as the basis. However, the formulation solves a different problem via non-linear optimization. Furgale *et al*. work has been extended to rolling shutter camera motion estimation [17] and calibration [15]. For structure from motion, Hedborg *et al*. [14] proposed a bundle adjustment (BA) scheme aware of the continuous camera motion, where pose is linearly interpolated across image rows and integrated into BA.

Tong *et al*. [23], based on Furgale *et al*. framework [13], proposed a non-parametric representation for SLAM, where the state is represented as a Gaussian process. This is shown to perform better than discrete-time SLAM, albeit with increased computation time. Another adaption of the framework to relative coordinates was recently presented by Anderson *et al*. [3]. Sheehan *et al*. [21] demonstrated a continuous trajectory localization algorithm from 3D data. The trajectory is represented with Catmull-Rom splines, and the method does not require correspondences. However, the method is nonlinear and requires a surveyed map.

Closest to our algorithm is the work presented by Bosse and Zlot [6]. Trajectory estimation is performed by deriving linear constraints from matched "surfels". The linear system is solved by iterative re-weighted least squares with linear interpolation of pose samples. Given an appropriate robot motion, they are able to match scans from a moving vehicle without requiring an additional pose sensor. For more complicated motions, a high grade IMU is required [7]. Establishing correspondences via surfel matching appears to increase the basin of convergence. However, surfels depend on the current shape of the point set and will need to be re-computed at every iteration, which is computationally demanding. In contrast, our method is more efficient operating on point-point correspondences only.

VII. Discussion

A. Choice of basis

The order of the basis functions affect the smoothness of the solution. If certain smoothness requirements are known/desired, then the order of the bases can be selected accordingly. For example, in motion planning, a trajectory is required to be smooth to minimize the jerk of the manipulator. In our case, we do not know the smoothness of the trajectory a-priori. However, since we know that the acceleration of the robot is bounded, the trajectory must be at least $\mathcal{C}^1$ continuous. Further, the robot traverses rough terrain and hence the trajectory is not "overly" smooth. Experimentally, we have found that quadratic and cubic B-spline bases work best for our data.

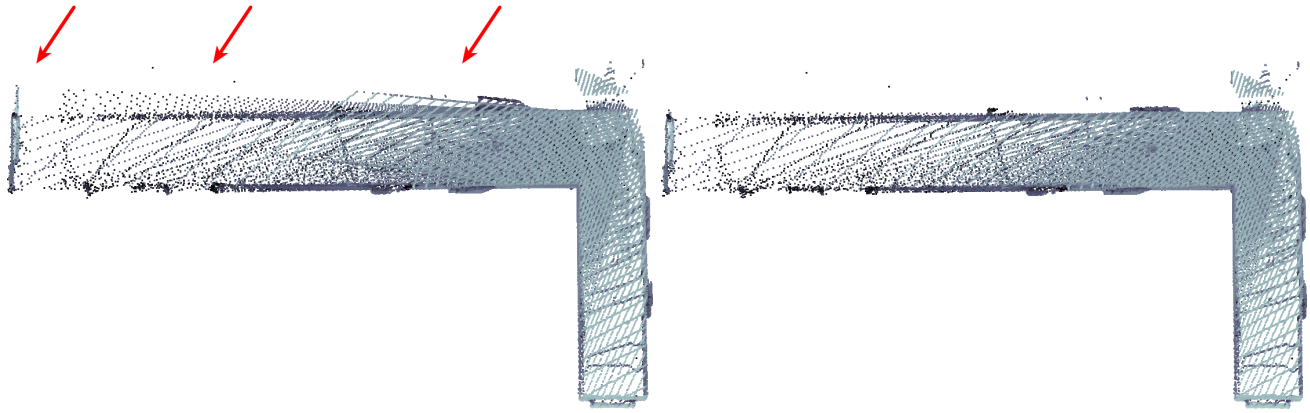

Fig. 7: Error in 3D structure due to VO dropouts for some parts of the data (left) cannot be correct with rigid registration methods. CICP (right) correctly accounts for such errors.

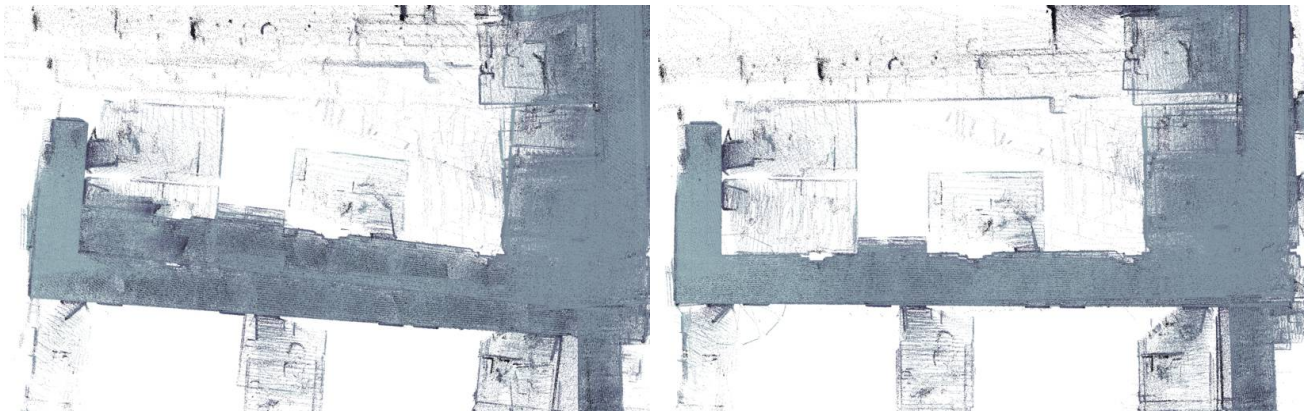

Fig. 8: Drift reduction over time with no explicit loop closure. On the left is the result with rigid GICP. Errors in frame undistortion accumulate over time and show visible mapping errors. CICP (right) eliminates much of these errors.

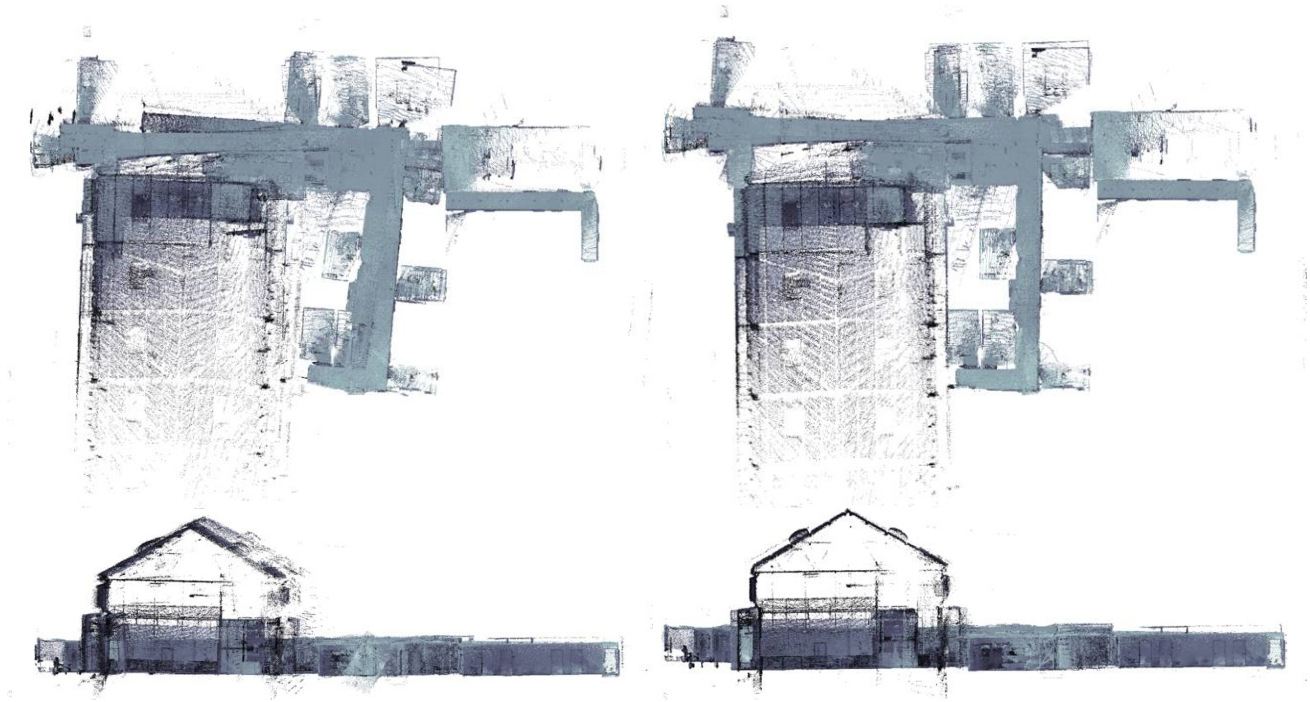

Fig. 9: Example of open-loop SLAM in a large indoor industrial environment.

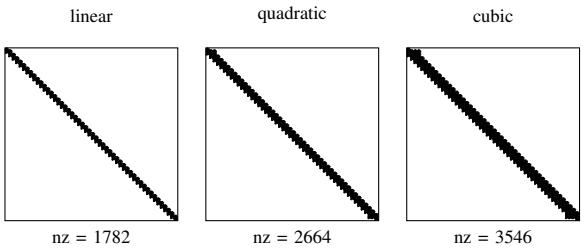

Fig. 10: Sparsity of the linear system in Eq.(9). The example shows 100 points with 50 poses using linear, quadratic and cubic basis functions. The matrix has 3×100 rows and 6×50 columns, dark areas indicate non-zero (nz) entries.

B. Computational efficiency & sparsity

Our algorithm is efficient as we only need to solve a linear system. Depending on the order of the basis functions (and their number) the resulting system is sparse. This is because the B-spline basis matrix is banded due to the compact support property. Fig.(10) shows a few examples of the sparsity pattern. Also, the basis function can be pre-computed offline when using uniform standardized intervals.

C. Limitations & failure cases

A linear solution is possible because we assume the existence of a stationary point set, which is held fixed. This might contribute to some of the long term drift of the system. It is possible to circumvent this by adjusting the registration scheme. For example, we may interchange the role of stationary vs. moving in the rolling buffer. This is to be explored in the future.

Mathematically, the algorithm has a singularity when the rotation angle is 180° as the Gibbs vector is undefined. However, this situation is not expected in practice as the robot does not change its heading instantaneously. Furthermore, when the algorithm is provided with an initialization, the rotation to be estimated is generally much less than 180°.

VIII. Conclusions and Future Work

In this work, we have presented an extension to classical ICP that is continuous-time trajectory estimation (CICP) algorithm. The trajectory was represented with a linear combination of B-spline basis functions producing a sparse linear system. The algorithm was tested on an open-loop 3D SLAM system from actuated lidar and was shown to perform better than rigid registration. In addition to the efficiency of solving the linear the system, we have demonstrated increased robustness of the method via L_1-regularization.

We plan to extend the algorithm for globally consistent mapping. Further, we plan on reducing initialization requirements by developing better methods for establishing correspondences to handle distortions of the data.

Acknowledgments

We would like to thank the anonymous reviewers for their careful read of an earlier draft and their valuable suggestions.

References

[1] H. Alismail, L. D. Baker, and B. Browning, "Automatic Calibration of a Range Sensor and Camera System," in *3DIMPVT*, 2012.

[2] H. Alismail, B. Browning, and M. B. Dias, "Evaluating Pose Estimation Methods for Stereo Visual Odometry on Robots," in *the 11th Int'l Conf. on Intelligent Autonomous Systems (IAS-11)*, 2010.

[3] S. Anderson and T. Barfoot, "Towards relative continuous-time SLAM," in *ICRA*, 2013, pp. 1033–1040.

[4] F. R. Bach, R. Jenatton, J. Mairal, and G. Obozinski, "Optimization with Sparsity-Inducing Penalties," *Foundations and Trends in Machine Learning*, vol. 4, no. 1, pp. 1–106, 2012.

[5] P. J. Besl and H. D. McKay, "A method for registration of 3-D shapes," *PAMI*, vol. 14, no. 2, pp. 239–256, 1992.

[6] M. Bosse and R. Zlot, "Continuous 3D scan-matching with a spinning 2D laser," in *ICRA*, 2009, pp. 4312–4319.

[7] M. Bosse, R. Zlot, and P. Flick, "Zebedee: Design of a Spring-Mounted 3-D Range Sensor with Application to Mobile Mapping," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1104–1119, 2012.

[8] S. Bouaziz, A. Tagliasacchi, and M. Pauly, "Sparse Iterative Closest Point," *Computer Graphics Forum*, vol. 32, no. 5, pp. 1–11, 2013.

[9] D. M. Cole and P. M. Newman, "Using laser range data for 3D SLAM in outdoor environments," in *ICRA*. IEEE, 2006, pp. 1556–1563.

[10] J. L. Crassidis and J. L. Junkins, *Optimal Estimation of Dynamic Systems*, 2nd ed. Chapman & Hall/CRC, 2011.

[11] C. Deboor, *A Practical Guide to Splines*. Springer, 1978.

[12] D. W. Eggert, A. Lorusso, and R. B. Fisher, "Estimating 3-D rigid body transformations: a comparison of four major algorithms," *Mach. Vision Appl.*, vol. 9, no. 5-6, pp. 272–290, Mar. 1997.

[13] P. T. Furgale, T. D. Barfoot, and G. Sibley, "Continuous-time batch estimation using temporal basis functions," in *ICRA*, 2012.

[14] J. Hedborg, P.-E. Forssen, M. Felsberg, and E. Ringaby, "Rolling shutter bundle adjustment," in *CVPR*, June 2012, pp. 1434–1441.

[15] Oth, L. and Furgale, P. and Kneip, L. and Siegwart, R., "Rolling shutter camera calibration," in *CVRP*, June 2013, pp. 1360–1367.

[16] K. Koh, S.-J. Kim, S. Boyd, and Y. Lin, "An interior-point method for large-scale L1-regularized logistic regression," *JMLR*, 2007.

[17] S. Lovegrove, A. Patron-Perez, and G. Sibley, "Spline Fusion: A continuous-time representation for visual-inertial fusion with application to rolling shutter cameras," *BMVC*, 2013.

[18] M. Muja and D. G. Lowe, "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration," in *Int'l Conf. on Computer Vision Theory and Application*, 2009, pp. 331–340.

[19] S. Rusinkiewicz and M. Levoy, "Efficient Variants of the ICP Algorithm," in *3DIM*, 2001.

[20] A. Segal, D. Haehnel, and S. Thrun, "Generalized-ICP," in *Proceedings of Robotics: Science and Systems*, June 2009.

[21] M. Sheehan, A. Harrison, and P. Newman, "Continuous vehicle localisation using sparse 3D sensing, kernelised Rényi distance and fast Gauss transforms," in *IROS*, 2013, pp. 398–405.

[22] G. Tam, Z.-Q. Cheng, Y.-K. Lai, F. Langbein, Y. Liu, D. Marshall, R. Martin, X.-F. Sun, and P. Rosin, "Registration of 3D Point Clouds and Meshes: A Survey from Rigid to Nonrigid," *IEEE Trans. on Vis. and Computer Graphics*, vol. 19, no. 7, pp. 1199–1217, 2013.

[23] C. H. Tong, P. Furgale, and T. D. Barfoot, "Gaussian Process Gauss-Newton for non-parametric simultaneous localization and mapping," *IJRR*, vol. 32, no. 5, pp. 507–525, 2013.

[24] G. Turk and M. Levoy, "Zippered Polygon Meshes from Range Images," in *SIGGRAPH*. ACM, 1994, pp. 311–318.

[25] M. Unser, "Splines: A perfect fit for signal and image processing," *Signal Processing Magazine, IEEE*, vol. 16, no. 6, pp. 22–38, 1999.