Asynchronous Multi-Sensor Fusion for 3D Mapping and Localization

Patrick Geneva, Kevin Eckenhoff, and Guoquan Huang

Abstract— **In this paper, we address the problem of 3D mapping and localization of autonomous vehicles while focusing on optimally fusing multiple heterogeneous and asynchronous sensors. To this end, based on the factor graph-based optimization framework, we design a modular sensor-fusion system that allows for efficient and accurate incorporation of any navigation sensor of different sampling rates. In particular, we develop a general method of out-of-sequence (asynchronous) measurement alignment to incorporate heterogeneous sensors into a factor graph for mapping and localization in 3D, without requiring the addition of new graph nodes, thus allowing the graph to have an overall reduced complexity. The proposed sensor-fusion system is validated on a collected dataset, in which the asynchronous-measurement alignment is shown to have an improved performance when compared to a naive approach without alignment.**

I. INTRODUCTION

Autonomous driving is an emerging technology that enables the reduction of traffic accidents and allows for those who are unable to drive for various medical conditions to regain their independence, by performing intelligent perception and planning based on multimodal sensors such as LIDARs, cameras, IMUs and GPS. It is critical for an autonomous vehicle to perform precise, robust localization for decision making as it is a sub-system that cannot fail during online autonomous operation. There have been a large amount of research efforts focused on multi-sensor fusion for state estimation for localization [20], which has reached a certain level of maturity, yielding a bounded problem given the well structured environment a vehicle operates in. In particular, graph-based optimization has recently prevailed for robot mapping and localization [2]. Due to the different sampling rates of the heterogeneous sensors, measurements arrive at different times. Accurate alignment of such out-of-sequence (i.e., asynchronous) measurements before optimally fusing them through graph optimization, while essential, has not been sufficiently investigated in the literature.[1]

Factor graph-based formulation [6] is desirable due to its ability to allow for the delayed incorporation of asynchronous measurements. Indelman et al. [11] address the problem of the inclusion of asynchronous measurements by taking advantage of IMU preintegrated terms. This allows them to incorporate any set of asynchronous sensors whose rates are longer than that of the IMU. Sünderhauf et al. [18] looked to address the incorporation of measurements with unknown time delays. Using high frequency odometry measurements, they create a state for *each* incoming odometry measurement so that delayed factors can be directly connected to its closest state. While both of these can be used to address arbitrary amounts of delay between sensors, they add a large amount of additional factors and edges to the graph. In contrast, the proposed approach incorporates measurements of different frequencies without significant increase of the overall graph complexity. It should be noted that while this does reduce the computational cost of optimization, reductions in graph size are always welcomed as a robot's physical memory becomes less of an issue.

Specifically, as the main contribution of this paper, we accurately align both asynchronous unary and binary graph factors based on our analytically derived linear 3D pose interpolation. This interpolation allows for the direct addition of asynchronous measurements into the graph, without the need for extra nodes to be added or for the naive ignoring of the measurement delay. Patron-Perez et al. [1] first proposed a spline-based trajectory method that allows for the fusion of delayed measurements with the consequence of an increase of overall system complexity and deviation from a pure pose graph. Outside of graph-based optimization, interpolation has been used to correct time offsets of continuous measurements such as LIDAR point clouds and rolling shutter cameras [3, 10]. In particular, Guo et al. [10] introduce the idea of linear interpolation between past camera poses, which allow for the use of extracted features from rolling shutter cameras. Ceriani et al. [3] use a linear interpolation between two poses in $SE(3)$ to unwarp lidar point measurements. In this work, we focus on the use of such linear interpolation in the graph-based optimization framework to allow for the efficient alignment of asynchronous measurements.

From the system perspective, we design and implement a modular framework for fusing a variety of sensors, where we separate the sensor fusion and pose estimation to allow for any sensor to be incorporated. This system design allows for the easy incorporation of additional sensors, while allowing for active sensor pose estimation modules to be changed without affecting the multi-sensor fusion. This is achieved by fusing emitted 3D pose estimates from sensor odometry (ego-motion) modules. The proposed sensor framework can then leverage these 3D poses, emitted in their own local frame of

This work was partially supported by the University of Delaware College of Engineering, UD Cybersecurity Initiative, the Delaware NASA/EPSCoR Seed Grant, the NSF (IIS-1566129), and the DTRA (HDTRA1-16-1-0039).

The authors are with the Department of Mechanical Engineering, University of Delaware, Newark, DE 19716, USA. Email: {pgeneva,keck,ghuang}@udel.edu

[1]It should be noted that the asynchronous measurement alignment under consideration is different from the time synchronization (or temporal sensor calibration) [19]; that is, even if sensors are well synchronized, their observations still arrive asynchronously.

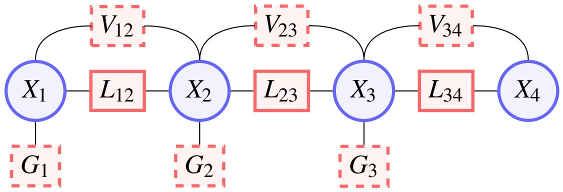

Fig. 1: Example factor graph that our system created. States that will be estimated are denoted in circles and measurements are denoted in squares. Note: we differentiate interpolated factors through the dashed outlines.

reference, in the global estimates of the robot.

II. GRAPH-BASED ESTIMATION

As the vehicle moves through the environment, a set of measurements, $\mathbf{z}$, is collected from its sensors, such as LIDAR scans, images, GPS, etc. These measurements relate to the underlying state to be estimated, $\mathbf{x}$. This process can be represented by a graph, where nodes correspond to parameters to be estimated (i.e., historical vehicle poses). Incoming measurements are represented as edges connecting their involved nodes (see Figure 1). Under the assumption of independent Gaussian noise corruption of our measurements, we formulate the Maximum Likelihood Estimation (MLE) problem as the following nonlinear least-squares problem [13]:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \sum_i ||\mathbf{r}_i(\mathbf{x})||^2_{\mathbf{P}_i} \tag{1}$$

Here, $\mathbf{r}_i$ is the zero-mean residual associated with measurement i, $\mathbf{P}_i$ is the measurement covariance, and $||\mathbf{v}||^2_{\mathbf{P}} = \mathbf{v}^\top \mathbf{P}^{-1} \mathbf{v}$ is the energy norm. This problem can be solved iteratively by linearizing about the current linearization point, $\hat{\mathbf{x}}^-$, and defining a new optimization problem in terms of the *error state*, $\Delta\mathbf{x}$:

$$\Delta\mathbf{x}^- = \arg\min_{\Delta\mathbf{x}} \sum_i \left|\left|\mathbf{r}_i\left(\hat{\mathbf{x}}^-\right) + \mathbf{H}_i \Delta\mathbf{x}\right|\right|^2_{\mathbf{P}_i} \tag{2}$$

Where $\mathbf{H}_i = \frac{\partial \mathbf{r}_i(\hat{\mathbf{x}}^- \boxplus \Delta\mathbf{x})}{\partial \Delta\mathbf{x}}$ is the Jacobian of i-th residual with respect to the error state. We define the generalized update operation, $\boxplus$, which maps a change in the error state to one in the full state. Given the error state $\{{}^i_G\tilde{\boldsymbol{\theta}}, {}^G\tilde{\mathbf{p}}_i\}$, this update operation can be written as $\{\mathrm{Expv}\left(-{}^i_G\tilde{\boldsymbol{\theta}}\right){}^i_G\mathbf{R}, {}^G\mathbf{p}_i + {}^G\tilde{\mathbf{p}}_i\}$.[2] After solving the linearized system, the current linearization point is updated as $\hat{\mathbf{x}}^+ = \hat{\mathbf{x}}^- \boxplus \Delta\mathbf{x}^-$. In this work, we parameterize the pose of each time step as $\{{}^i_G\mathbf{R}, {}^G\mathbf{p}_i\}$, which describes the rotation from the global frame $\{G\}$ to the local frame $\{i\}$ and the position of the frame $\{i\}$ seen from the global frame $\{G\}$ of reference. This linearization process is then repeated until convergence. While there are openly available solvers [5, 12, 13], the computational complexity of the graph based optimization can reach $O(n^3)$ in the worst case.

[2] Throughout the paper we denote the vector form of the matrix exponential as $Expv(\cdot)$ which maps $\mathbb{R}^3 \rightarrow SO(3)$, (e.g. $\mathrm{Expv}({}^G_I\boldsymbol{\theta}) = {}^G_I\boldsymbol{R}$). Similarly, we define the matrix logarithm $Logv(\cdot)$ to map between $SO(3) \rightarrow \mathbb{R}^3$, (e.g. $\mathrm{Logv}({}^G_I\boldsymbol{R}) = {}^G_I\boldsymbol{\theta}$).

A reduction in the number of states being estimated can both help with the overall computational complexity and the physical size of a graph during long term SLAM. Naively, if a new node is to be created at each sensor measurement time instance, the overall graph optimization frequency can suffer. To prolong high frequency graph optimization, we present our novel method of measurement alignment which allows for the estimation of the poses of a *single* sensor's measurements.

III. ASYNCHRONOUS MEASUREMENT ALIGNMENT

A. Unary Factors

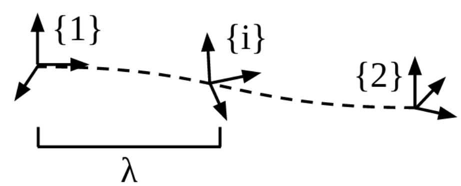

Fig. 2: Given two measurements in the global frame of reference $\{1\}$ and $\{2\}$, we interpolate to a new pose $\{i\}$. The above λ is the time-distance fraction that defines how much to interpolate the pose.

Unary factors can appear when sensors measure information in respect to a single node. For example, GPS can provide global position measurements indirectly through latitude, longitude, and altitude readings, while LIDAR scan-matching to known maps can provide a direct reading of the global pose. Motivated to not add new graph nodes when receiving asynchronous data, we add a "corrected" measurement to an existing node by preforming pose interpolation between two sequential sensor measurements. Note: that for GPS measurements we only need to perform 3D position interpolation, however for completeness we have derived the following interpolation for a 3D pose. We define a time-distance fraction between two consecutive poses as follows:

$$\lambda = \frac{(t_i - t_1)}{(t_2 - t_1)} \tag{3}$$

where t_1 and t_2 are the timestamps of the bounding measurements, and t_i is the desired interpolation time (i.e. the timestamp of the existing node). Under the assumption of a constant velocity motion model, we interpolate between the two pose readings:

$${}^i_G\mathbf{R} = \mathrm{Expv}\left(\lambda\ \mathrm{Logv}({}^2_G\mathbf{R}{}^1_G\mathbf{R}^\top)\right){}^1_G\mathbf{R} \tag{4}$$

$${}^G\mathbf{p}_i = (1-\lambda){}^G\mathbf{p}_1 + \lambda{}^G\mathbf{p}_2 \tag{5}$$

where $\{{}^i_G\boldsymbol{R}, {}^G\boldsymbol{p}_i\}$ is the interpolated measurement 3D pose and $\{{}^1_G\boldsymbol{R}, {}^G\boldsymbol{p}_1\}$ and $\{{}^2_G\boldsymbol{R}, {}^G\boldsymbol{p}_2\}$ are the bounding poses. While this interpolated measurement can now be directly added to the graph, the last step is to correctly compute the corresponding covariance needed in graph-based optimization.

Hence, we perform the following covariance propagation:

$$\boldsymbol{P}_i = \boldsymbol{H}_u \boldsymbol{P}_{1,2} \boldsymbol{H}_u{}^T \tag{6}$$

$$\boldsymbol{H}_u = \begin{bmatrix} \frac{\partial_G^i \tilde{\boldsymbol{\theta}}}{\partial_G^1 \tilde{\boldsymbol{\theta}}} & \mathbf{0}_{3\times3} & \frac{\partial_G^i \tilde{\boldsymbol{\theta}}}{\partial_G^2 \tilde{\boldsymbol{\theta}}} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \frac{\partial^G \tilde{\boldsymbol{p}}_i}{\partial^G \tilde{\boldsymbol{p}}_1} & \mathbf{0}_{3\times3} & \frac{\partial^G \tilde{\boldsymbol{p}}_i}{\partial^G \tilde{\boldsymbol{p}}_2} \end{bmatrix} \tag{7}$$

where $\boldsymbol{P}_{1,2}$ is the joint covariance matrix from the bounding poses, and $\tilde{\boldsymbol{\theta}}$ and $\tilde{\boldsymbol{p}}$ are the error states of each angle and position measurement, respectively. For detailed calculations of all Jacobians derived in this paper, we refer the reader to the companion tech report [9]. The resulting non-zero Jacobian matrix entries are defined as:

$$\frac{\partial_G^i \tilde{\boldsymbol{\theta}}}{\partial_G^1 \tilde{\boldsymbol{\theta}}} = -{}_1^i\boldsymbol{R} \left(J_r\left(\lambda \ \mathrm{Logv}({}_1^2\boldsymbol{R})\right) \ \lambda \ J_r^{-1}\left(\mathrm{Logv}({}_1^2\boldsymbol{R})\right) - I \right) \tag{8}$$

$$\frac{\partial_G^i \tilde{\boldsymbol{\theta}}}{\partial_G^2 \tilde{\boldsymbol{\theta}}} = {}_1^i\boldsymbol{R} \ J_r\left(-\lambda \ \mathrm{Logv}({}_1^2\mathbf{R}^\top)\right) \ \lambda \ J_r^{-1}\left(\mathrm{Logv}({}_1^2\mathbf{R}^\top)\right) \tag{9}$$

$$\frac{\partial^G \tilde{\boldsymbol{p}}_i}{\partial^G \tilde{\boldsymbol{p}}_1} = (1-\lambda)\ \mathbf{I} \ , \quad \frac{\partial^G \tilde{\boldsymbol{p}}_i}{\partial^G \tilde{\boldsymbol{p}}_2} = \lambda \ \mathbf{I} \tag{10}$$

where the Right Jacobian of $SO(3)$ denoted as $J_r(\phi)$ and its inverse $J_r^{-1}(\phi)$ is defined as the following [4, 8]:

$$J_r(\phi) = \mathbf{I} - \frac{1-cos(\parallel \phi \parallel)}{\parallel \phi \parallel^2} \lfloor \phi \times \rfloor + \frac{\parallel \phi \parallel - sin(\parallel \phi \parallel)}{\parallel \phi \parallel^3} \lfloor \phi \times \rfloor^2 \tag{11}$$

$$J_r^{-1}(\phi) = \mathbf{I} + \frac{1}{2} \lfloor \phi \times \rfloor + \left(\frac{1}{\parallel \phi \parallel^2} - \frac{1+cos(\parallel \phi \parallel)}{2 \parallel \phi \parallel sin(\parallel \phi \parallel)} \right) \lfloor \phi \times \rfloor^2 \tag{12}$$

B. Binary Factors

Designing multi-sensor systems for estimation often requires fusing asynchronous odometry readings from different sensor modules (e.g., ORB-SLAM2 [14] or LOAM [21]). A difficulty that arises is the unknown transformation between the global frame of references of each module. This unknown comes from both the rigid transformation between sensors (which can be found through extrinsic calibration) and each module *initializes* its global frame of reference independently. Rather than directly modifying the codebase of each module, we combine the sequential odometry measurements into *relative* transforms; thereby, we remove the ambiguity of each module-to-module transformation.

In particular, given two poses in the second sensor's world frame, $\{{}_o^1\boldsymbol{R}, {}^o\boldsymbol{p}_1\}$ and $\{{}_o^2\boldsymbol{R}, {}^o\boldsymbol{p}_2\}$ with the joint covariance $\boldsymbol{P}_{1,2}$, we calculate the relative transformation as follows:

$${}_1^2\boldsymbol{R} = {}_o^2\boldsymbol{R} {}_o^1\boldsymbol{R}^\top \tag{13}$$

$${}^1\boldsymbol{p}_2 = {}_o^1\boldsymbol{R}\left({}^o\boldsymbol{p}_2 - {}^o\boldsymbol{p}_1\right) \tag{14}$$

where we define the unknown global frame of these 3D pose measurements as $\{o\}$ and their corresponding reference frames as $\{1\}$ and $\{2\}$. To calculate the relative covariance matrix, we perform the following covariance propagation based on the above measurement transformation:

$$\boldsymbol{P}_{12} = \boldsymbol{H}_r \boldsymbol{P}_{1,2} \boldsymbol{H}_r{}^T \tag{15}$$

where $\boldsymbol{P}_{1,2}$ is the joint covariance matrix of each pose in the $\{o\}$ frame of reference. The resulting Jacobian matrix $\boldsymbol{H}_r$ is defined as the following:

$$\boldsymbol{H}_r = \begin{bmatrix} -{}_1^2\boldsymbol{R} & \mathbf{0}_{3\times3} & \boldsymbol{I}_{3\times3} & \mathbf{0}_{3\times3} \\ \lfloor {}_o^1\boldsymbol{R}({}^o\boldsymbol{p}_2 - {}^o\boldsymbol{p}_1)\times \rfloor & -{}_o^1\boldsymbol{R} & \mathbf{0}_{3\times3} & {}_o^1\boldsymbol{R} \end{bmatrix} \tag{16}$$

We now have the $\{{}_1^2\boldsymbol{R}, {}^1\boldsymbol{p}_2\}$ relative transform between two poses and corresponding covariance $\boldsymbol{P}_{12}$. If this transformation is not in the same sensor frame of reference (e.g., relative transform is between camera to camera and the state is LIDAR to LIDAR), one can use the method described in Appendix A to convert the measurement into the frame of reference of the state.

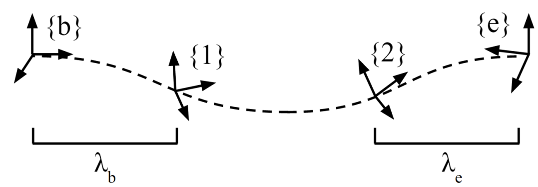

Fig. 3: Given a relative transformation, calculated using (13) and (14), between the $\{1\}$ and $\{2\}$ frame of reference, we extrapolate this relative transformation to the desired beginning $\{b\}$ and end $\{e\}$ poses. The above λs are the time-distance fractions that we use to extrapolate the relative transformation.

Due to the asynchronous nature of the measurements from two different sensors, the times corresponding to the beginning and end of the relative transformation will not align with matched existing state poses. Therefore, under the assumption of a constant velocity motion, we *extrapolate* the relative transformation across the desired interval. This intuitively corresponds to a "stretching" of the relative pose measurement in time. We define two time-distance fractions that determine how much the relative transformation needs to be extended (see Figure 3):

$$\lambda_b = \frac{t_1 - t_b}{t_2 - t_1} \qquad \lambda_e = \frac{t_e - t_2}{t_2 - t_1} \tag{17}$$

The λ's describe the magnitude that the relative transformation is to be "stretched" in each direction, with the subscripts b and e denoting the beginning and end state poses. These time-distance fractions can also be negative, corresponding to the "shrinking" of the relative transformation. Given the relative transform and the time-distance fractions, we define the following extrapolation equations:

$${}_b^e\boldsymbol{R} = \mathrm{Expv}\left[(1+\lambda_b+\lambda_e)\mathrm{Logv}\left({}_1^2\boldsymbol{R}\right)\right] \tag{18}$$

$${}^b\boldsymbol{p}_e = (1+\lambda_b+\lambda_e)\mathrm{Expv}\left[-\lambda_b \mathrm{Logv}\left({}_1^2\boldsymbol{R}\right)\right] {}^1\boldsymbol{p}_2 \tag{19}$$

The covariance propagation is then given by:

$$\boldsymbol{P}_{be} = \boldsymbol{H}_i \boldsymbol{P}_{12} \boldsymbol{H}_i{}^\top \ , \quad \boldsymbol{H}_i = \begin{bmatrix} \frac{\partial_b^e \tilde{\boldsymbol{\theta}}}{\partial_1^2 \tilde{\boldsymbol{\theta}}} & \mathbf{0}_{3\times3} \\ \frac{\partial^b \tilde{\boldsymbol{p}}_e}{\partial_1^2 \tilde{\boldsymbol{\theta}}} & \frac{\partial^b \tilde{\boldsymbol{p}}_e}{\partial^1 \tilde{\boldsymbol{p}}_2} \end{bmatrix} \tag{20}$$

where $\boldsymbol{P}_{12}$ is the relative factor covariance calculated above. The resulting non-zero Jacobian matrix entries are defined as:

$$\frac{\partial_b^e \tilde{\boldsymbol{\theta}}}{\partial_1^2 \tilde{\boldsymbol{\theta}}} = J_r \left[(1+\lambda_b+\lambda_e)\mathrm{Logv}({}_1^2\boldsymbol{R}^\top)\right](1+\lambda_b+\lambda_e)J_r^{-1}\left[\mathrm{Logv}({}_1^2\boldsymbol{R}^\top)\right] \tag{21}$$

$$\frac{\partial^b \tilde{\boldsymbol{p}}_e}{\partial_1^2 \tilde{\boldsymbol{\theta}}} = \Big(-(1+\lambda_b+\lambda_e)\mathrm{Expv}\left[\lambda_b \mathrm{Logv}({}_1^2\boldsymbol{R}^\top)\right]\lfloor {}^1\hat{\boldsymbol{p}}_2\times\rfloor \\ J_r(\lambda_b \mathrm{Logv}({}_1^2\boldsymbol{R}^\top))\lambda_b J_r^{-1}(\mathrm{Logv}({}_1^2\boldsymbol{R}^\top))\Big) \tag{22}$$

$$\frac{\partial^b \tilde{\boldsymbol{p}}_e}{\partial^1 \tilde{\boldsymbol{p}}_2} = (1+\lambda_b+\lambda_e)\mathrm{Expv}\left[-\lambda_b \mathrm{Logv}\left({}_1^2\boldsymbol{R}\right)\right] \tag{23}$$

IV. SYSTEM DESIGN

A. Design Motivations

The proposed method allows for the reduction of the overall graph complexity during asynchronous sensor fusion. We now propose a system that leverages the use of asynchronous sensors in the application of autonomous driving. To both facilitate the flexibility of the vehicle design and reduce cost, we aim to run the system on a vehicle *without* access to a GPS unit and with low cost asynchronous sensors (i.e., without the use of electronic triggering). This design constraint presents the unique challenge of still needing to localize the vehicle in the GPS frame of reference without the use of a traditional GPS sensor. By publishing the vehicle state estimate in the GPS frame of reference, we allow for existing global path planning and routing modules to continue to work as expected. To overcome this challenge, we present a unique prior LIDAR map that allows for the vehicle to both initialize and localize in the GPS frame of reference. Specifically we design a framework with two separate sub-systems as follows:

- Creation of an accurate prior map using a vehicle that has an additional Real Time Kinematic (RTK) GPS sensor unit.
- Leverage the prior map in GPS denied localization to determine the 3D pose in the GPS frame of reference.

This framework is flexible and cost effective as only a single "collection" vehicle is needed to build the prior map that multiple lower cost vehicles can leverage. Specifically, this prior map allows for localization in the GPS frame of reference without the use of GPS measurements during runtime and can support localization in GPS denied environments (e.g., tunnels or parking garages). Both sub-systems can leverage the proposed asynchronous factor interpolation to enable the use of low cost asynchronous sensors while ensuring a reduction of overall graph complexity.

B. System Overview - Prior Map

The first sub-system we propose is one that generates an accurate prior map that can be leveraged by the second sub-system to localize in the GPS frame of reference. Shown in Figure 4, we fuse odometry measurements from openly available stereo and LIDAR modules, ORB-SLAM2 [14] and LOAM [21], respectively, with a RTK GPS unit. Both of these modules provide six degree of freedom pose estimation.[3][4] We estimate LIDAR states connected with consecutive non-interpolated binary factors from LOAM LIDAR odometry. To provide additional robustness and information into the graph, we connect consecutive states with interpolated binary factors (Section III-B) from ORB-SLAM2 visual odometry. To ensure that the estimated states are in the GPS frame of reference, we attach interpolated unary factors (Section III-A) from the RTK GPS sensor. Both ORB-SLAM2 visual binary factors and RTK GPS unary factors need to be interpolated because both sensors are asynchronous to the LIDAR sensor.

The graph can be solved in real-time using an incremental solver such as iSAM2 [12] or offline with a full batch solver. It is then simple to construct a prior map using the estimated states and their corresponding LIDAR point clouds. To evaluate the overall quality of the generated prior map point cloud, the cloud is visually inspected for misalignment on environmental planes such as walls or exterior of buildings. The generated prior map from the experimental dataset can be see in Figure 5.

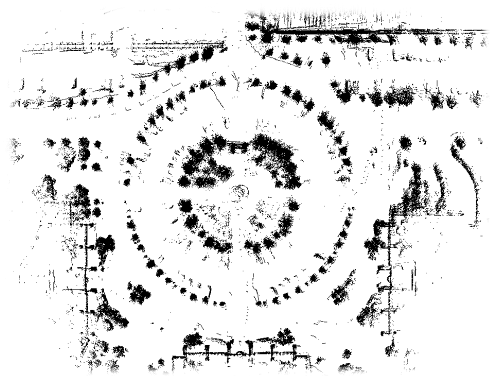

Fig. 5: Prior map generated from the experimental dataset

C. System Overview - GPS Denied Localization

Using the generated prior map, localization in the GPS frame can be preformed *without* the use of a GPS sensor. As seen in Figure 4, we estimate LIDAR states that are connected with non-interpolated and interpolated binary factors (Section III-B) from LOAM and ORB-SLAM2 odometry modules, respectively. In addition to these two binary factors, we preform Iterative Closest Point (ICP) matching between the newest LIDAR point cloud to the generated prior map. This ICP transform can then be added as a non-interpolated unary factor into the factor graph. These unary factors constrain the graph to be in the GPS frame of reference during 3D pose estimation.

To provide real-time localization capabilities, we leverage the iSAM2 solver during GPS denied state estimation. The

[3]Note: both modules do *not* normally provide a corresponding covariance needed for batch optimization. We reference the reader to the appendices in the companion tech report [9].

[4]We run LOAM in the default mode, while ORB-SLAM2 is run first in "mapping" mode to generate a map of the environment. We then use ORB-SLAM2 in "localization" mode to ensure that the estimate does not jump due to loop closures when fusing its output with the proposed systems.

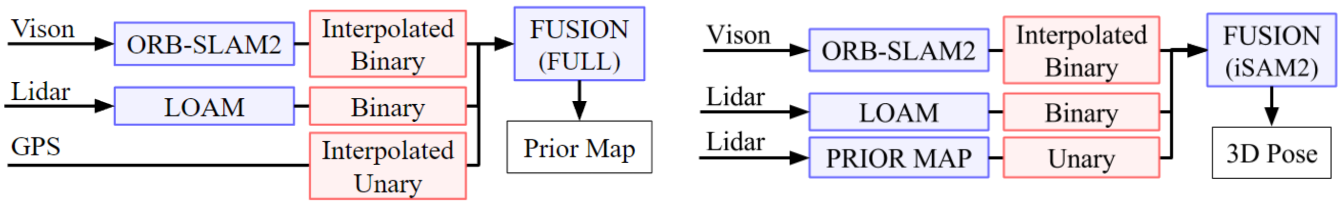

Fig. 4: Overview of the flow of data through the system, where all incoming measurements are denoted on the far left of each figure. These measurements are first processed through an odometry module if needed (seen in blue) and then converted into factors (seen in red) that can be added to factor graph. The prior map system (left) leverages RTK GPS measurements to create a prior map in the GPS frame of reference. The GPS denied estimation system (right) uses the generated LIDAR maps to ensure that the pose estimation is in the GPS frame of reference.

estimation operates at the frequency of the LIDAR sensor limited only by the speed the LOAM module can process measurements. It was found that when creating a unary factor using ICP matching to the prior map took upwards of 1-2 seconds. To overcome this long computation time, incoming LIDAR clouds are processed at a lower frequency in a secondary thread, and then added to the graph after successful ICP matching.

V. EXPERIMENTAL RESULTS

A. System Validation

To access the overall performance of the GPS denied system, we constructed a data collection vehicle with both a combination of low cost sensors and a RTK GPS sensor. The vehicle is equipped with a 8 channel Quanergy M8 LIDAR [16], ZED stereo camera [17], and RTK enabled NovAtel Propak6 GPS sensor [15]. The Quanergy M8 LIDAR was run at 10Hz, while the ZED stereo camera was run at 30Hz with a resolution of 672 by 376. The RTK enabled NovAtel Propak6 GPS sensor operated at 20Hz with an average accuracy of ± 15 centimeters. The GPS solution accuracy allows for the creation of a high quality prior map (see Figure 5). To facilitate the GPS denied system, a dataset was first collected on the vehicle and then processed using a full batch solver. Following the proposed procedure in Section IV-B, LIDAR factors are added to the factor graph, while both stereo and GPS factors are interpolated and then directly connected to corresponding LIDAR states. The resulting LIDAR point cloud, created in the GPS frame of reference, can then be used during GPS denied navigation.

To represent the real world, the GPS denied system was tested on the day following the data collection for the prior map. This was to introduce changes in the environment, such as changes in car placement and shrubbery, while also showing that the prior map can still be leveraged. The same vehicle was used with the only difference being that the RTK GPS was not used in the GPS denied localization. This RTK GPS was instead used to provide an accurate ground truth comparison. Following the proposed procedure in Section IV-C, incoming LIDAR point clouds are matched to the map generated the previous day and then added to the factor graph after successful ICP alignment.

The estimated vehicle state is compared to the corresponding output of the RTK GPS. As seen in Figure 6, when performing GPS denied localization, the system was able to remain within a stable 2 meter accuracy.

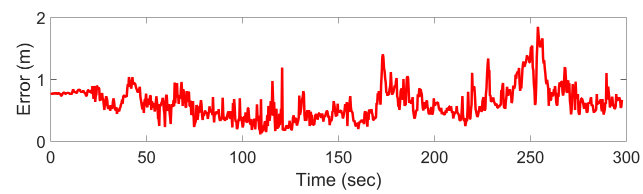

Fig. 6: Average position error magnitude over 10 runs. GPS denied estimation compared at each time instance, of the 840 meter long run, with the RTK GPS position. Average vehicle speed of 6mph.

B. Evaluating the Asynchronous Measurement Alignment

Having shown that the system is able to accurately localize in real-time without the use of GPS, we next evaluated how the interpolation impacts the estimation. To do so, we did not use the ICP matching to the LIDAR prior cloud and instead only used the pure odometry from LOAM and ORB-SLAM2. We compared the proposed factor interpolation method against a naive approach of factor addition into the graph which ignores the issue of time delay and directly attaches incoming factors to the closest nodes without interpolation.

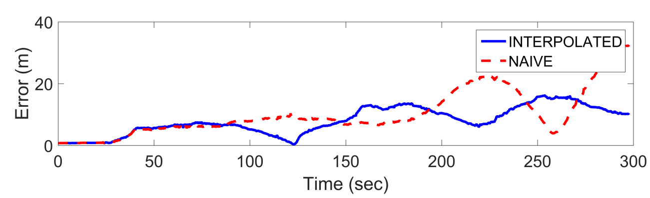

Fig. 7: Comparison of the proposed method and a naive approach of adding incoming factors to the closest nodes, denoted as “interpolation” and “naive” respectively.

Seen in Figure 7, the proposed factor interpolation outperformed the estimation accuracy of the naive approach. The *average* error of the naive approach was 10.24 meters and the proposed method’s average error is 8.00 meters (overall

21.8% decrease). This shows that the use of interpolation on incoming binary factors can greatly increase the estimation accuracy, without increasing graph complexity.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we have developed a general approach of asynchronous measurement alignment within the graph-based optimization framework of mapping and localization in order for optimal fusion of multimodal sensors. The designed framework provides a modular system with the ability to replace individual modules and allow for any sensor to be incorporated. The system has been tested on a experimental dataset and compared to a naive approach to show the improvement due to the proposed asynchronous measurement alignment. Looking forward, we will incorporate other sensors, such as Inertial Measurement Units (IMUs), through the use of IMU preintegration developed in our prior work [7] to improve the system fault tolerance, if the main sensor fails. We will also investigate how to improve the current mapping and localization, in particular, when autonomously driving in dynamic urban environments.

APPENDIX

A. Static Transformations

Another issue that commonly arises in the application of multi-sensor fusion is the ability to convert from one frame of reference to another. For example, in this paper, binary factors from the external ORB-SLAM2 visual odometry library. Given this relative transform $\{{}^{C2}_{C1}\boldsymbol{R}, {}^{C1}\boldsymbol{p}_{C2}\}$ in the camera frame and a corresponding covariance $\boldsymbol{P}_{C12}$, we would like to transform from the camera to the LIDAR sensor frame. This can be done as follows:

$$ {}^{L2}_{L1}\boldsymbol{R} = {}^{L}_{C}\boldsymbol{R}\, {}^{C2}_{C1}\boldsymbol{R}\, {}^{L}_{C}\boldsymbol{R}^{\top} \tag{24} $$

$$ {}^{L1}\boldsymbol{p}_{L2} = {}^{L}_{C}\boldsymbol{R} \left({}^{C2}_{C1}\boldsymbol{R}^{\top}\, {}^{C}\boldsymbol{p}_{L} + {}^{C1}\boldsymbol{p}_{C2} - {}^{C}\boldsymbol{p}_{L} \right) \tag{25} $$

where we define the LIDAR frame of reference as $\{Li\}$, $i \in \{1,2\}$ and the camera frame of reference as $\{Ci\}$, $i \in \{1,2\}$. It is assumed that the static transform, $\{{}^{L}_{C}\boldsymbol{R}, {}^{C}\boldsymbol{p}_{L}\}$, from the LIDAR to camera frame of reference are known from offline calibration. Given the above transform, special care needs to be taken to calculate the relative covariance matrix in the LIDAR frame of reference as follows:

$$ \boldsymbol{P}_{L12} = \boldsymbol{H}_s \boldsymbol{P}_{C12} \boldsymbol{H}_s{}^{\top} \tag{26} $$

where $\boldsymbol{P}_{C12}$ is the relative camera covariance. For detailed calculations of this Jacobian, please see the companion tech report [9]. The resulting Jacobian matrix $\boldsymbol{H}_s$ is defined as the following:

$$ \boldsymbol{H}_s = \begin{bmatrix} {}^{L}_{C}\boldsymbol{R} & \boldsymbol{0}_{3\times3} \\ -{}^{L}_{C}\boldsymbol{R}\, {}^{C2}_{C1}\boldsymbol{R}^{\top} \lfloor {}^{C}\boldsymbol{p}_{L} \times \rfloor & {}^{L}_{C}\boldsymbol{R} \end{bmatrix} \tag{27} $$

REFERENCES

[1] Steven Lovegrove Alonso Patron-Perez and Gabe Sibley. "A Spline-Based Trajectory Representation for Sensor Fusion and Rolling Shutter Cameras". In: *International Journal of Computer Vision* 113.3 (2015), pp. 208–219. ISSN: 1573-1405. DOI: 10.1007/s11263-015-0811-3.

[2] Cesar Cadena et al. "Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age". In: *IEEE Transactions on Robotics* 32.6 (2016), pp. 1309–1332.

[3] Simone Ceriani et al. "Pose interpolation SLAM for large maps using moving 3D sensors". In: *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE. 2015, pp. 750–757.

[4] Gregory S Chirikjian. *Stochastic Models, Information Theory, and Lie Groups, Volume 2: Analytic Methods and Modern Applications*. Vol. 2. Springer Science & Business Media, 2011.

[5] Frank Dellaert. "Factor graphs and GTSAM: A hands-on introduction". In: (2012).

[6] Frank Dellaert and M. Kaess. "Square Root SAM: Simultaneous Localization and Mapping via Square Root Information Smoothing". In: *International Journal of Robotics Research* 25.12 (Dec. 2006), pp. 1181–1203.

[7] Kevin Eckenhoff, Patrick Geneva, and Guoquan Huang. "High-Accuracy Preintegration for Visual-Inertial Navigation". In: *Proc. of International Workshop on the Algorithmic Foundations of Robotics*. San Francisco, CA, Dec. 2016.

[8] Christian Forster et al. "IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation". In: *Robotics: Science and Systems*. Georgia Institute of Technology. 2015.

[9] Patrick Geneva, Kevin Eckenhoff, and Guoquan Huang. *Asynchronous Multi-Sensor Fusion for 3D Mapping and Localization*. Tech. rep. RPNG-2017-002. Available: http://udel.edu/~ghuang/papers/tr_async.pdf. University of Delaware, 2017.

[10] Chao X Guo et al. "Efficient Visual-Inertial Navigation using a Rolling-Shutter Camera with Inaccurate Timestamps." In: *Robotics: Science and Systems*. Citeseer. 2014.

[11] Vadim Indelman et al. "Information fusion in navigation systems via factor graph based incremental smoothing". In: *Robotics and Autonomous Systems* 61.8 (2013), pp. 721–738.

[12] M. Kaess et al. "iSAM2: Incremental smoothing and mapping with fluid relinearization and incremental variable reordering". In: *2011 IEEE International Conference on Robotics and Automation*. May 2011, pp. 3281–3288. DOI: 10.1109/ICRA.2011.5979641.

[13] R. Kummerle et al. "g2o: A general framework for graph optimization". In: *Proc. of the IEEE International Conference on Robotics and Automation*. Shanghai, China, May 2011, pp. 3607–3613.

[14] Raul Mur-Artal and Juan D Tardos. "ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras". In: *arXiv preprint arXiv:1610.06475* (2016).

[15] Novatel. *ProPak6 Triple-Frequency GNSS Receiver*. https://www.novatel.com/assets/Documents/Papers/ProPak6-PS-D18297.pdf.

[16] Inc. Quanergy Systems. *Quanergy M8 LIDAR*. http://quanergy.com/m8/.

[17] Stereolabs. *ZED Stereo Camera*. https://www.stereolabs.com/zed/specs/.

[18] Niko Sünderhauf, Sven Lange, and Peter Protzel. "Incremental Sensor Fusion in Factor Graphs with Unknown Delays". In: *Advanced Space Technologies in Robotics and Automation (ASTRA)*. 2013.

[19] Z. Taylor and J. Nieto. "Motion-Based Calibration of Multimodal Sensor Extrinsics and Timing Offset Estimation". In: *IEEE Transactions on Robotics* 32.5 (Oct. 2016), pp. 1215–1229.

[20] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. MIT Press, 2005.

[21] Ji Zhang and Sanjiv Singh. "LOAM: Lidar Odometry and Mapping in Real-time." In: *Robotics: Science and Systems*. Vol. 2. 2014.