INCREMENTAL SENSOR FUSION IN FACTOR GRAPHS WITH UNKNOWN DELAYS

Niko Sünderhauf, Sven Lange, and Peter Protzel

Department of Electrical Engineering and Information Technology
Chemnitz University of Technology, 09111 Chemnitz, Germany.

ABSTRACT

Sensor fusion by incremental smoothing in factor graphs allows the easy incorporation of asynchronous and delayed measurements, which is one of the main advantages of this approach compared to the ubiquitous filtering techniques. While incorporating delayed measurements into the factor graph representation is in principle easy when the delay is known, handling *unknown* delays is a non-trivial task that has not been explored before in this context. Our paper addresses the problem of performing incremental sensor fusion in factor graphs when some of the sensor information arrive with a significant unknown delay. We develop and compare two techniques to handle such delayed measurements under mild conditions on the characteristics of that delay: We consider the unknown delay to be bounded and quantizable into multiples of the state transition cycle time. The proposed methods are evaluated using a simulation of a dynamic 3-DoF system that fuses odometry and GPS measurements.

1. INTRODUCTION

While Bayesian filtering techniques like the extended or unscented Kalman filter are unarguably the method of choice for many applications of sensor fusion, there are certain situations where incremental *smoothing* approaches based on inference in factor graphs perform significantly better. Especially in the case of highly non-linear systems, we can expect such optimization-based smoothing approaches to yield more accurate results than filtering methods, due to the incremental nature of the optimization that allows repeated linearization. A second advantage is that *delayed* sensor messages can be incorporated very easily. Proper processing of such delayed information is of importance in applications involving decentralized sensor networks, or in vehicle-to-vehicle, or vehicle-to-infrastructure communication in the automotive or aerospace domain. All these systems have to incorporate information from external sensors that have to be transmitted over a communication channel where varying delays have to be expected.

Filtering approaches have difficulties in correctly incorporating such delayed measurements, since they only keep track of the most recent state estimate and have to filter backwards in time, correct the prediction with the new sensor data and then filter forwards again. This problem has been addressed early in the literature e.g. by the classical forward-backward-smoother of Rauch, Tung and Striebel [2], but is still under consideration today, as recent papers like [3, 4, 5] show. In a *factor graph*-based framework, the process is much more straightforward. Incorporating that newly arrived, but delayed sensor data corresponds to inserting an additional node along with its factors into the factor graph representation of the problem, which can be done efficiently online [6].

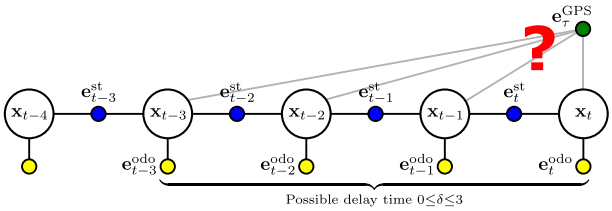

Figure 1: The problem addressed in this paper: The state of a vehicle $\mathbf{x}_t$ is estimated from frequent odometry measurements (yellow), a state transition including a motion model (blue), and comparatively rare GPS measurements (green). This task is solved within a factor graph framework, applying incremental inference using the iSAM2 [1] approach. At time t, a new GPS measurement arrives, but the sensor measurement was delayed by an unknown time $0 \leq \delta_t \leq \delta_{\max}$. It is not clear how the new factor (green) can be incorporated into the factor graph, since it is unknown to which state variable it should be connected.

Approaches for efficient inference in Gaussian factor graphs have been successfully applied to solve large-scale SLAM problems (Simultaneous Localization and Mapping, e.g. [7, 8]) for several years now. It has been only recently that advances in the field of *incremental* inference in such graphs led to the first demonstrations of an adaptive-lag smoother for incremental sensor fusion based on factor graphs. This work by Indelman et al. [9] builds upon the iSAM2 algorithm [1] that uses a special data structure, called the Bayes tree, to perform efficient incremental smoothing in an adaptive-lag manner.

Although [9] and [6] demonstrated the incremental fu-

sion of asynchronous, delayed sensors in factor graphs, the exact delays of the measurements were assumed to be known. Often however, the measurement delay will be *unknown* to the system, or at least only *uncertain* knowledge will exist. This situation occurs e.g. when the sensor data is communicated over a channel that can not guarantee or specify the transmission lag, or when the data is preprocessed and thus delayed before any synchronized timestamping occurred. Often, sensors do not support proper time synchronization with the host that performs the sensor fusion or the host runs a non-real-time operating system and cannot reliably timestamp the received data. In any case, it is not known with certainty, when a measurement $\mathbf{z}_t^\tau$ received at time t was really taken by the sensor. This unknown measurement time τ is given as $\tau = t - \delta$ where δ is the unknown or uncertain delay we are interested in, as illustrated in Fig. 1. If not handled properly, these delays will cause systematic errors in the estimated variables.

Although the problem of sensor fusion with uncertain or unknown time delays has been addressed in the context of filtering before (e.g. [10, 11, 12, 13]), to the best of our knowledge the problem has not yet been considered in the context of graph-based sensor fusion.

We start by reviewing the basics of factor graph-based inference in the next section. In the following we consider three different approaches of how delayed measurements can be incorporated into a factor graph for incremental sensor fusion if the delay δ is unknown. We will then evaluate and compare these approaches using a simulation of a 3-DoF vehicle.

2. FACTOR GRAPHS FOR PROBABILISTIC ESTIMATION

In a general probabilistic estimation problem, we are interested in the distribution over a set of state variables $\mathcal{X}$, given measurements or prior knowledge $\mathcal{Z}$. In other words, we want to know the conditional probability distribution $P(\mathcal{X}|\mathcal{Z})$. Such a distribution can be *factored* into a product expression

$$P(\mathcal{X}|\mathcal{Z}) = \prod_i P_i(\bar{\mathcal{X}}_i|\bar{\mathcal{Z}}_i) \tag{1}$$

where $\bar{\mathcal{X}}_i \subseteq \mathcal{X}$ and $\bar{\mathcal{Z}}_i \subseteq \mathcal{Z}$ are subsets of $\mathcal{X}$ and $\mathcal{Z}$ respectively, according to the dependency structure between the hidden variables $\mathcal{X}$ and the given evidence $\mathcal{Z}$ (e.g. measurements or a-priori knowledge). Such a factorization can be expressed with a *factor graph*.

Factor graphs are bipartite undirected graphs and have been proposed by [14] as a general tool to model factorizations of large functions with many variables into smaller local subsets. Since factor graphs are bipartite by definition, they contain two sets of nodes: one for the hidden variables and the other for the probabilistic relations (the factors) between them.

Consider the example in Fig. 1. For convenience, the variable nodes are always illustrated larger than the (colored) factor nodes. Notice how a factor can connect two variables, like the state transition factors illustrated in blue. Other factors are unary, i.e. they represent sensor measurements or prior knowledge, like the yellow odometry factors or the green GPS-position factor.

2.1. Finding the Maximum a Posteriori Solution

The maximum a posteriori (MAP) estimate of the distribution $P(\mathcal{X}|\mathcal{Z})$, i.e. the most likely variable configuration $\mathcal{X}^*$ given the data $\mathcal{Z}$, is formalized as an optimization problem of the form

$$\mathcal{X}^* = \operatorname*{argmax}_{\mathcal{X}} P(\mathcal{X}|\mathcal{Z}) = \operatorname*{argmax}_{\mathcal{X}} \prod_i P_i(\bar{\mathcal{X}}_i|\bar{\mathcal{Z}}_i) \tag{2}$$

If the single factors P_i are Gaussian, they are of the general form

$$P_i(\bar{\mathcal{X}}_i|\bar{\mathcal{Z}}_i) = \eta \exp -\frac{1}{2}\|\mathbf{e}_i(\bar{\mathcal{X}}_i, \bar{\mathcal{Z}}_i)\|^2_{\mathbf{\Sigma}_i} \tag{3}$$

where $\mathbf{e}_i(\bar{\mathcal{X}}_i, \bar{\mathcal{Z}}_i)$ is a problem-specific error function. Notice the three different types of specific error functions in the graph depicted in Fig. 1 that contain error models for state transition ($\mathbf{e}^{\text{st}}$), odometry measurements ($\mathbf{e}^{\text{odo}}$), and GPS position measurements ($\mathbf{e}^{\text{GPS}}$).

Using relation (3) and taking the negative logarithm, we can transform (2) into

$$\mathcal{X}^* = \operatorname*{argmin}_{\mathcal{X}} \sum_i \|\mathbf{e}_i(\bar{\mathcal{X}}_i, \bar{\mathcal{Z}}_i)\|^2_{\mathbf{\Sigma}_i} \tag{4}$$

which is a least squares optimization problem, since we seek the minimum over a sum of squared terms.

Such problems can be solved using a variety of methods like Levenberg-Marquardt, Gauss-Newton or Powell's Dog-Leg. These approaches iteratively solve the problem by repeatedly linearizing it and updating the current estimate of $\mathcal{X}^*$ until convergence. At their heart, these methods rely on a factorization (either QR or Cholesky) of the Jacobian associated with the factor graph. Specialized solvers that exploit the sparse nature of the factorization (i.e. the sparse structure of the Jacobians) can solve typical problems with thousands of variables very efficiently. Examples for convenient C++ frameworks that contain such solvers and can be easily applied to a number of different problem domains are g^{2}o [8] or GTSAM [15].

2.2. iSAM2 - Incremental Inference in Factor Graphs

In general, factor graph problems can be either solved in *batch* mode or *incrementally*. The difference is that a

batch solver uses all available measurements and solves the complete graph at once. In contrast, incremental methods are able to efficiently update the graph (i.e. incorporate new measurements) and calculate a new estimate online, after each update step. Such incremental solvers have been explored by Kaess et al. who introduced the iSAM [7] algorithm (*incremental smoothing and mapping*) and more recently iSAM2 [1].

Their key insight was that the sparse QR or Cholesky factorization that lies at the heart of batch solvers like Levenberg-Marquardt or Gauss-Newton is equivalent to converting the factor graph into a Bayes net via an *elimination* algorithm. The resulting Bayes net can be further converted into a new data structure coined the *Bayes tree* [16] by discovering the cliques in the Bayes net. This tree structure allows particularly easy and efficient incremental updates and inference. That is, new factors can be added and a new updated estimate $\mathcal{X}^*$ is calculated where only the necessary parts of the tree are re-evaluated.

For more details we have to refer the reader to [1] which gives an elaborate description of iSAM2 and the Bayes tree.

3. INCORPORATING MEASUREMENTS WITH UNKNOWN DELAY IN FACTOR GRAPHS

After the short review of factor graph-based inference in the previous section, we now address the core topic of our paper and explore how delayed measurements can be incorporated into the factor graph framework when the delay is not exactly known. We will first set out the considered scenario before proposing a solution in the following.

We consider a dynamical system with system state $\mathbf{x}_t = (x, y, \theta, v, \omega)^\mathsf{T}$ that evolves over time by following a CTRV (Constant Turn Rate and Velocity) model. An odometry sensor generates measurements $\mathbf{u}_t = (v, \omega)^\mathsf{T}$ with a measurement rate of $1/\Delta_{\text{Odo}}$. Therefore, every time a new odometry measurement $\mathbf{u}_t$ arrives, we generate a new state variable $\mathbf{x}_t$ and connect it to the odometry measurement factor and to its predecessor $\mathbf{x}_{t-\Delta_{\text{Odo}}}$ via a CTRV state transition factor. Both factors are illustrated in Fig. 1, where the state transition is depicted as a blue node, while the odometry measurement factors are shown in yellow. The same color code will be used in the following figures.

In addition to the odometry measurements, we receive absolute position measurements $\mathbf{z}_t^\tau = (x, y)^\mathsf{T}$ at a slower rate $1/\Delta_{\text{GPS}}$. Such a measurement $\mathbf{z}_t^\tau$ is received at time t but was created by the sensor at time $\tau = t - \delta_t$. That is, it was delayed by a lag δ_t, before it was received and timestamped by the sensor fusion subsystem. Notice that we denote the delay δ_t with a time index since the delay may not be constant over time.

Let us now assume for a moment that the delay δ_t is known to the system. In this case, we can express the optimal (maximum a posteriori, MAP) configuration of system states X conditioned on the odometry measurements U and position measurements Z to be

$$X^* = \underset{X}{\operatorname{argmax}}\, P(X|U, Z) \tag{5}$$

The conditional probability $P(X|U, Z)$ can be factored into

$$P(X|U, Z) = \prod_t P(\mathbf{x}_t|\mathbf{u}_t) \cdot P(\mathbf{x}_t|\mathbf{x}_{t-1}) \cdot \prod_\tau P(\mathbf{x}_\tau|\mathbf{z}_\tau) \tag{6}$$

The MAP estimate is then found by taking the negative log and assuming all distributions to be Gaussians:

$$X^* = \underset{X}{\operatorname{argmin}} \sum_t \|\mathbf{e}_t^{\text{odo}}\|^2_{\mathbf{\Sigma}} + \sum_t \|\mathbf{e}_t^{\text{st}}\|^2_{\mathbf{\Lambda}} + \sum_\tau \|\mathbf{e}_\tau^{\text{GPS}}\|^2_{\mathbf{\Omega}} \tag{7}$$

This is a nonlinear least squares optimization problem that we solve and update incrementally using iSAM2 [1].

The error function of the state transition factor is given by

$$\mathbf{e}_t^{\text{st}} = f^{\text{CTRV}}(\mathbf{x}_{t-1}) - \mathbf{x}_t \tag{8}$$

where f^{CTRV} is the motion model.The odometry factor cost function is simply

$$\mathbf{e}_t^{\text{odo}} = \mathbf{x}_t^{[v,\omega]} - \mathbf{u}_t \tag{9}$$

where $\mathbf{x}_t^{[v,\omega]} = (v_t, \omega_t)^\mathsf{T}$. Similar, the GPS factor cost function is defined as

$$\mathbf{e}_\tau^{\text{GPS}} = \mathbf{x}_\tau^{[x,y]} - \mathbf{z}_t^\tau \tag{10}$$

While the MAP estimation problem in (7) can be solved easily and efficiently by incremental solvers like iSAM2 or batch solvers like g^{2}o, it is non-trivial to set up the factor graph when the exact value of the delay δ_t is unknown. As was illustrated in Fig. 1, it is unclear to which state variable the measurement factor should be connected.

Let us consider that only the maximum possible delay time δ_{max} is known, and that δ_t can be assumed to be quantized to a multiple of the state transition cycle time Δ_{st}. So formally it is known that

$$\delta_t = n \cdot \Delta_{\text{st}} \mid 0 \leq \delta_t \leq \delta_{\text{max}}, n \in \mathbb{Z} \tag{11}$$

The quantization assumption is valid if Δ_{st} is small. In many applications, internal sensor measurements (e.g. IMU or odometry) will arrive with high rates. Even if such high-frequency measurements are not available, it is always possible to create new state variables at arbitrary high rates, as long as a state transition function can be defined over arbitrary short time intervals. We therefore see that the quantization assumption does not pose severe restrictions on the scenario.

In the following, we will consider the state transition cycle time Δ_{st} to be equal to Δ_{Odo}, the odometry rate.

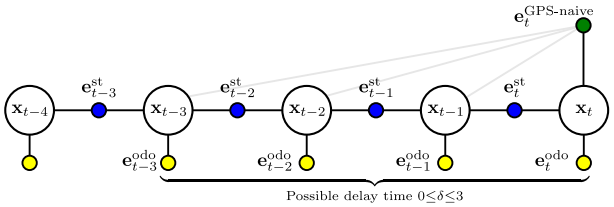

Figure 2: The naive approach: A new measurement $\mathbf{z}_t^\tau$ (green) arrives at time t and is simply connected with the newest variable node $\mathbf{x}_t$, ignoring the significant possible delay.

In the remainder of this section, we describe three possible approaches of incorporating the delayed measurements $\mathbf{z}_t^\tau$ into the factor-graph-based sensor fusion system. All three strategies will be compared and evaluated using results from a simulation in the next section.

3.1. The Naive Approach – Ignoring the Delay

Of course the simplest strategy is to ignore the possible delay δ_t and treat all measurements $\mathbf{z}_t^\tau$ as if they had been created at time t instead of τ. This naive approach would simply associate the incoming new measurement $\mathbf{z}_t^\tau$ with the newest state node $\mathbf{x}_t$, as illustrated in Fig. 2. Since in reality it should be attached to $\mathbf{x}_\tau$ instead, we can expect large systematic errors when applying this naive method.

The error function of the GPS factor for the naive approach is defined as:

$$\mathbf{e}_t^{\text{GPS-naive}} = \mathbf{x}_t^{[x,y]} - \mathbf{z}_t^\tau \tag{12}$$

3.2. Maximum Likelihood Selection

Another strategy is to perform a maximum likelihood selection procedure. Together with every measurement $\mathbf{z}_t^\tau$, a set of candidate system states $\mathcal{X}_t = \{\mathbf{x}_{t-\delta_{\max}}, \ldots, \mathbf{x}_t\}$ is maintained. This set consists of all the state variables, the measurement $\mathbf{z}_t^\tau$ could possibly be associated to in reality, since we know $t - \delta_{\max} \leq \tau \leq t$. In the following, the index γ is used to denote the timestamp of one of these state variables, therefore $\gamma = t \ldots t - \delta_{\max}$.

Whenever the measurement factor containing $\mathbf{z}_t^\tau$ is to be evaluated, we evaluate it for all $\mathbf{x}_\gamma \in \mathcal{X}_t$ and return the result with the lowest error. Thus we select

$$\mathbf{x}_\gamma^* = \underset{\mathbf{x}_\gamma \in \mathcal{X}_t}{\operatorname{argmin}} \, \| \mathbf{x}_\gamma^{[x,y]} - \mathbf{z}_t^\tau \|_{\mathbf{\Omega}}^2 \tag{13}$$

and treat the factor as if it was connected solely to $\mathbf{x}_\gamma^*$. Therefore, the resulting cost function of the GPS factor is

$$\mathbf{e}_t^{\text{GPS-MLS}} = \mathbf{x}_\gamma^{*[x,y]} - \mathbf{z}_t^\tau \tag{14}$$

This selection approach is inspired from the max-mixture models of [17]. An illustration is given in Fig. 3.

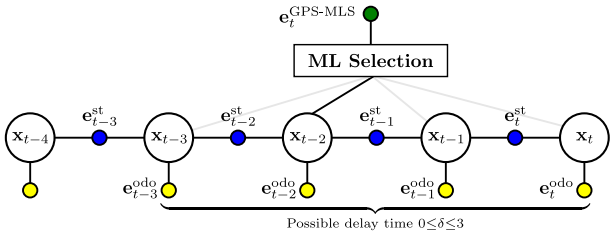

Figure 3: Maximum likelihood selection: The delayed measurement $\mathbf{z}_t^\tau$ (green) is dynamically connected to the state variable that locally maximizes the measurement's likelihood. The selection process is performed whenever the factor is to be evaluated, but afterwards the factor is treated as if it was only connected to the selected state variable.

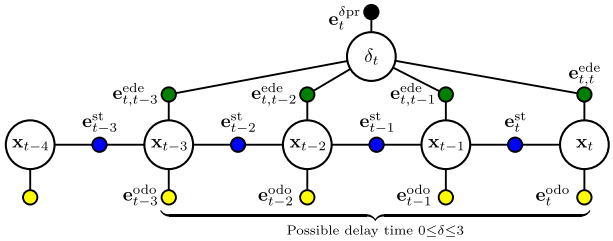

Figure 4: The explicit delay estimation approach: The unknown delay δ_t is explicitly estimated for each arriving measurement $\mathbf{z}_t^\tau$. The measurement arrives at time t but is estimated to originate at time $\tau = t - \delta_t$. Therefore, the measurement factors (green) are weighted according to the difference between the estimated measurement time τ and the timestamp of the associated variable vertex $\mathbf{x}_\gamma$.

3.3. Explicit Delay Estimation

The most promising approach in terms of expected accuracy is to explicitly *estimate* the delay δ_t for each measurement $\mathbf{z}_t^\tau$ individually. To achieve this, we add a latent variable for the unknown δ_t to the factor graph when a new measurement $\mathbf{z}_t^\tau$ arrives.

As in the ML selection approach described above, we know that this new measurement $\mathbf{z}_t^\tau$ could be connected to any one of the N state variables from the set $\mathcal{X}_t = \{\mathbf{x}_{t-\delta_{\max}}, \ldots, \mathbf{x}_t\}$. But instead of *selecting* one of these states like above, we rather create N new factors $\phi_{t,\gamma}^{\text{ede}}$. Each of these factors connects one of the N state variables from $\mathcal{X}_t$ to the newly introduced delay variable for δ_t. Fig. 4 illustrates this concept and shows the factors $\phi_{t,\gamma}^{\text{ede}}$ in green. The superscript $^{\text{ede}}$ stands for *explicit delay estimation*.

The error function of such a factor $\phi_{t,\gamma}^{\text{ede}}$ is defined as

$$\mathbf{e}_{t,\gamma}^{\text{ede}} = w \cdot \left(f_{[x,y]}^{\text{CTRV}}(\mathbf{x}_\gamma, (t - \delta_t) - \gamma) - \mathbf{z}_t^\tau \right) \tag{15}$$

where $(t - \delta_t) - \gamma$ is the difference between the timestamp γ of the state variable $\mathbf{x}_\gamma$ and the currently estimated temporal origin of the measurement $\mathbf{z}_t^\tau$. The function $f_{[x,y]}^{\text{CTRV}}$

is the motion model that predicts how the state of $\mathbf{x}_\gamma$ evolves during the given time interval. In other words, we use that motion model to predict the system state at the time $t - \delta_t$, which is our current best estimate of the origin of $\mathbf{z}_t^\tau$. Notice that $f_{[x,y]}^{\text{CTRV}}$ only returns the $(x, y)^\mathsf{T}$ part of the full five dimensional state vector.

The weight factor w in (15) is defined as a squared exponential:

$$w = \exp -\frac{1}{2}\left(\frac{(t-\delta_t)-\gamma}{\sigma_\text{w}}\right)^2 \tag{16}$$

This term is close to 1 if the estimated origin τ of $\mathbf{z}_t^\tau$ is close to the timestamp of the regarded state variable $\mathbf{x}_\gamma$ and drops towards 0 otherwise. The denominator σ_w controls how quickly the weight falls off and has been set empirically to $\sigma_\text{w} = \frac{1}{3}\Delta_\text{Odo}$.

Intuitively, one can understand this weight w as a kind of *switch*, that *gradually* removes factors from the graph. Of the N factors $\phi_{t,\gamma}^\text{ede}$ (e.g. the four green factors in Fig. 4), ideally only those are "active" that connect to a state variable $\mathbf{x}_\gamma$ so that γ is close to the current estimate of the temporal origin of the delayed measurement $\mathbf{z}_t^\tau$. In other words, depending on the current estimate of the delay δ_t, the weights w will automatically "connect" $\mathbf{z}_t^\tau$ to the correct state variable $\mathbf{x}_\gamma$.

This whole mechanism shares strong resemblance with our previous work on *switchable constraints* [18] in the context of robust pose graph SLAM. This special formulation also used weights to remove or weaken the influence of certain factors from the graph.

An additional prior factor for the delay variable with error function $\mathbf{e}_t^{\delta\text{pr}}$ ensures the validity of the estimate of δ_t as defined by (11). If necessary, it is also possible to model a random drift of the delay δ_t over time. Combining the previous definitions, we gain the problem formulation for the explicit delay estimation approach, which is:

$$\begin{aligned} X^*, \Delta^* = \quad & \text{argmin}_{X,\Delta} \textstyle\sum_t \|\mathbf{e}_t^\text{odo}\|_\mathbf{\Sigma}^2 + \sum_t \|\mathbf{e}_t^\text{st}\|_\mathbf{\Lambda}^2 \\ & + \textstyle\sum_t \sum_\gamma \|\mathbf{e}_{t,\gamma}^\text{ede}\|_\mathbf{\Omega}^2 + \sum_t \|\mathbf{e}_t^{\delta\text{pr}}\|_\Xi^2 \end{aligned} \tag{17}$$

In the last line we of course only sum over the valid times t where a GPS measurement $\mathbf{z}_t^\tau$ was received. Notice that we collected all δ_t into the set Δ and that the optimization problem is now defined over two sets of variables.

4. EXPERIMENTS AND RESULTS

We evaluated and compared the performance of the different strategies to deal with the unknown delay in a simulation. This section explains this simulation and presents the evaluation results.

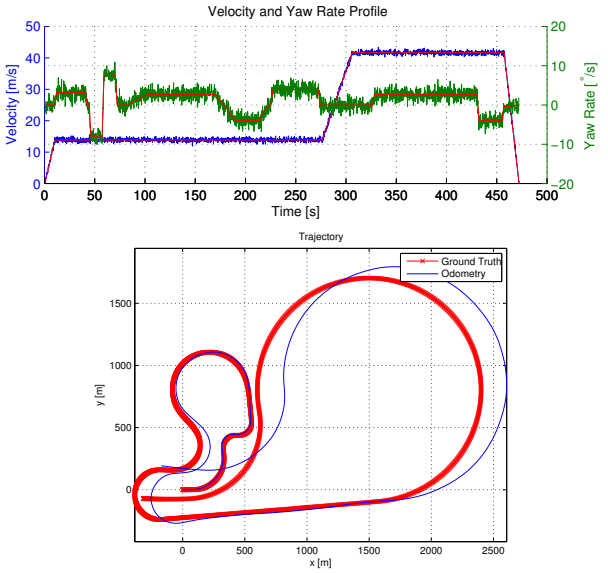

Figure 5: Top: Noisy measurements (green / blue) and ground truth (red) for velocity and yaw rate used in the simulation. Bottom: The resulting ground truth trajectory and the trajectory estimated from the noisy odometry.

Table 1: Parameters of the Simulation

Parameter	Value
Δ_Odo	0.2 s
Δ_GPS	0.2 s . . . 7.8 s
δ_max	$\min(\Delta_\text{GPS}, 2.0\,\text{s})$
σ_w	$\frac{1}{3}\Delta_\text{Odo}$

4.1. Description of the Simulation

We simulated a vehicle in a 5 dimensional state space $\mathbf{x}_t = (x, y, \theta, v, \omega)^\mathsf{T}$ for a total simulation time of 472 seconds. In this simulation, the vehicle followed the velocity and yaw rate profiles depicted in Fig. 5. Since the odometry measurements $(v, \omega)^\mathsf{T}$ were subject to noise, the trajectory according to the odometry alone substantially deviates from the ground truth trajectory, as can be seen in the bottom part of Fig. 5.

As laid out before, odometry measurements $\mathbf{u}_t = (v, \omega)^\mathsf{T}$ arrived with cycle time $\Delta_\text{Odo} = 0.2\,\text{s}$, creating 2360 measurements and state variables $\mathbf{x}_t$ during the 472 simulated seconds. In contrast, the position measurements $\mathbf{z}_t^\tau = (x, y)^\mathsf{T}$ were created every Δ_GPS seconds. To examine the influence of the GPS measurement frequency, we varied Δ_GPS in steps of 0.4 seconds between 0.2 and 7.8 seconds. For each value of Δ_GPS, we conducted 100 complete trials of the simulation.

Furthermore, the GPS measurements arrived with a delay δ_t that was unknown to the system. We did not keep the

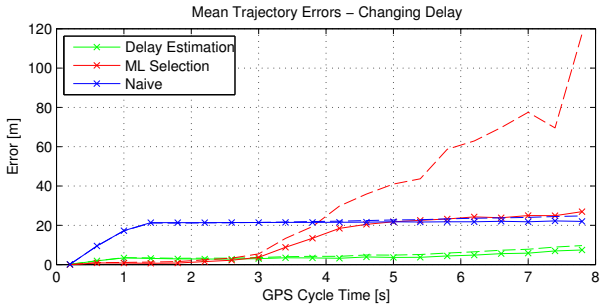

Figure 6: Mean trajectory errors vs. simulated GPS cycle time Δ_{GPS} for the different strategies and a *changing* delay δ_t. The value of δ_t changed over time, see Fig. 7. The dashed lines illustrates the errors of the concurrent estimates while the solid lines are the errors after final smoothing.

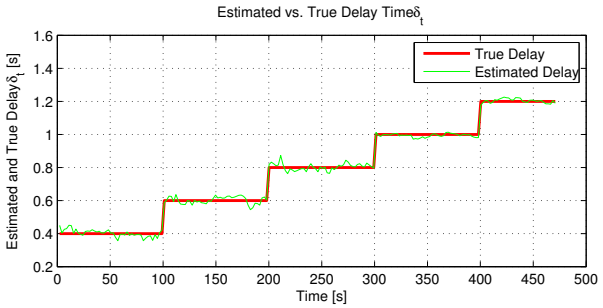

Figure 7: Changing delay: The delay δ_t changed during the simulation and increased from 0.4 to 1.2 seconds. The plot compares the true delay (red) with an exemplary estimation result from the explicit delay estimation method (green).

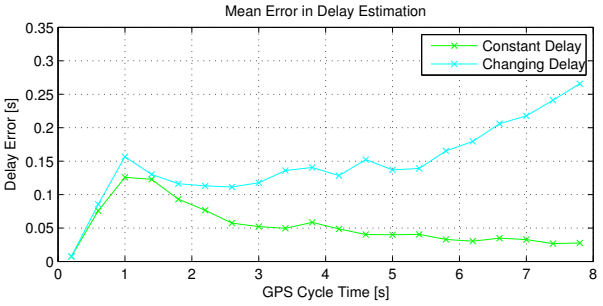

Figure 8: Mean errors between the estimated delays δ_t (estimated by the explicit delay estimation approach) and the true delays for the constant and changing delay experiments.

unknown delay time δ_t constant during one run of the simulation, but rather let it change over time: Starting from a delay of 0.4 seconds, the delay increased by 0.2 seconds every 100 seconds. This results in a maximum delay time of 1.2 seconds.

To compare the performance of the three proposed strategies (naive, maximum likelihood selection, and explicit delay estimation), we implemented the necessary factors for iSAM2 [1] and used it to perform incremental sensor fusion. The fused trajectory was compared to the ground truth trajectory and the euclidean errors were calculated to indicate the quality of the chosen approach. Notice that we distinguish between "trajectory errors" and "*concurrent* trajectory errors". The first uses the estimated trajectory after final smoothing, whereas the latter one compares the estimated vehicle positions after each timestep, i.e. using the newest estimated $\mathbf{x}_t$ after it has been appended to the factor graph and the `update` operation of iSAM2 has been performed. The concurrent error is in a way more meaningful, since it represents the adaptive-lag smoothing results available during *online* operation in contrast to *batch* operation after all data has been gathered.

4.2. Results and Discussions of the Experiments

The resulting mean trajectory errors over all trials for each of the three proposed methods are compared in Fig. 6. The naive strategy of ignoring the unknown delay time δ_t obviously produces high systematic errors in the resulting estimated trajectory. We can see the blue curve increases with growing GPS cycle time Δ_{GPS} until the value of $\delta_{\text{max}} = 1.2\,\text{s}$ is reached. Then it levels off and stays almost constant at a value of approximately 20 meters. This behaviour is predictable, because of how the delay time δ_t is chosen and bounded. The maximum delay time δ_t will increase until $\Delta_{\text{GPS}} = \delta_{\text{max}}$ and will then remain constant with further increasing Δ_{GPS}. The concurrent errors and the errors after final smoothing behave almost identical, only at very large Δ_{GPS} can we see a small relative increase of the concurrent error.

In contrast to the naive strategy, the maximum likelihood selection (shown in red) and the delay estimation method (green) produce very low and comparable errors until $\Delta_{\text{GPS}} \approx 3\,\text{s}$. After that point, the trajectory errors for the ML selection approach increase sharply. The errors after final smoothing reach a level that is identical to the naive strategy. In contrast, the concurrent errors increase well beyond this level, reaching mean values of over 100 meters in the end. This indicates that the maximum likelihood selection strategy is unsuitable for online data fusion in this experiment for large Δ_{GPS}.

The best results were achieved by the delay estimation approach. Its trajectory errors remain low over the complete span of Δ_{GPS}. This is true for both the concurrent errors as for the errors after final smoothing. As expected, the concurrent errors are slightly higher. Apart from the low trajectory errors shown in Fig. 6, the proposed approach of explicit delay estimation was also able to maintain a good estimate of the actual delay δ_t. Fig. 7 illustrates how the delay δ_t changed over time and shows an exemplary estimation result.

In addition to the experiments where the delay δ_t changed over time, we repeated the simulations but held the de-

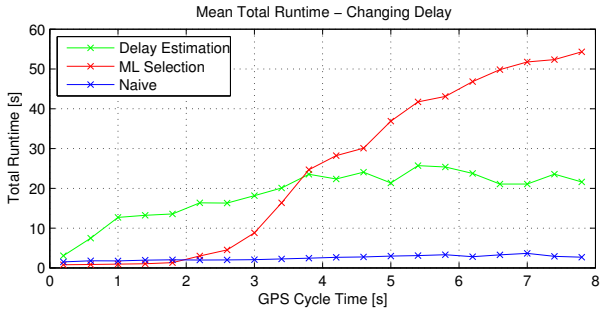

Figure 9: Comparison of the mean total runtime for the three different analyzed strategies. All experiments were conducted on an Intel Core i7-2600 CPU running at 3.4 GHz, using iSAM2 from the GTSAM package.

lay constant over time (initialized with a random value between 0 and $\delta_{\max}$). As expected, the obtained results were very similar, but the unknown delay could be estimated more accurately. Fig. 8 compares the errors of the estimated delay for the changing and constant delay experiments.

The explanation for the inferior behaviour of the MLS approach is that when the GPS measurements are available only rarely (i.e. Δ_{GPS} is large), the locally most likely state variable picked by the ML selection strategy is not necessarily the correct one anymore. The longer the time intervals Δ_{GPS} between the GPS measurements get, the more will the odometry-based trajectory estimates diverge from the real trajectory. At some point, the ML-selection scheme starts to associate the GPS measurements $\mathbf{z}_t^\tau$ with the wrong state variables $\mathbf{x}_\gamma$, causing systematic errors in the overall estimation. Since the results after final smoothing are much lower, these effects seem to be mitigated by the smoother over time as more data becomes available.

Apart from the high estimation errors, the maximum likelihood selection scheme is not reliable in its convergence. The solver (iSAM2, with Gauss-Newton) started to diverge for many of the trials, with a clear correlation between divergence probability and Δ_{GPS}.

Regarding the total runtime, the maximum likelihood selection approach is superior to the delay estimation method until $\Delta_{\text{GPS}} > 3\,\text{s}$ (see Fig. 9). After this point the MLS method becomes practicably unusable due to the increasing runtime, the divergence and the high estimation errors.

5. CONCLUSIONS

Our paper analyzed how measurements with unknown delays can be best incorporated into a graph-based incremental sensor fusion framework. Our experiments showed that of the three compared approaches, the only feasible method is to incorporate the measurements by applying the explicit delay estimation approach proposed in section 3.3. This method explicitly estimates the unknown delay δ_t for each measurement and would also allow to incorporate prior knowledge or a drift model of δ_t.

As expected, the naive approach of completely ignoring the delay resulted in high systematic errors in the state estimation. The maximum likelihood selection (MLS) strategy produced promising results at first, but is unreliable since it exhibits divergent behaviour when the intervals between the delayed measurements get larger. In this case, the errors accumulated by the noisy other sensors (e.g. odometry in our simulation) cause MLS to pick the wrong state node since the locally most likely association is not the best on a global level. Depending on the actual system and sensor setup, MLS might be feasible if the uncertainties of state prediction are low or correcting measurements arrive at a comparably high rate. It seems that MLS is the more likely to degrade the more uncertain the state prediction gets during single instances of delayed measurements.

Overall, the work presented in this paper is the result of a first attempt to address the problem of sensor fusion with delayed measurements in factor graphs when the delay is unknown. Unarguably, a more elaborate examination of the problem is necessary. The chosen scenario of fusing absolute position with odometry measurements on a CTRV vehicle might be too limited to conclude that the presented explicit delay estimation approach is feasible in *all* possible scenarios of sensor fusion.

Also our assumptions made on the delay characteristics need to be examined closely. While the boundedness and the quantization assumption appear to be relatively mild, the fact that the delay stays constant for longer periods of time needs to be given close scrutiny. Is the proposed approach still valid when the delay varies from measurement to measurement, e.g. following a stochastic process?

In future work we therefore want to conduct more simulations and also evaluate the proposed method on real data collected in different scenarios, e.g. fusing IMU and delayed vision information to estimate the state of a highly dynamic quadrotor UAV.

REFERENCES

1. M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert. iSAM2: Incremental Smoothing and Mapping Using the Bayes Tree. *Intl. Journal of Robotics Research*, 2012.

2. H. E. Rauch, C. T. Striebel, and F. Tung. Maximum Likelihood Estimates of Linear Dynamic Systems. *Journal of the American Institute of Aeronautics and Astronautics*, 3(8):1445–1450, August 1965.

3. Y. Bar-Shalom. Update with Out-of-Sequence Measurements in Tracking: Exact Solution. *Aerospace and Electronic Systems, IEEE Transactions on*, 38(3):769 – 777, jul 2002.

4. Marc Peter Deisenroth and Henrik Ohlsson. A General Perspective on Gaussian Filtering and Smoothing: Explaining Current and Deriving New Algorithms. In *Proceedings of the 2011 American Control Conference (ACC)*, 2011.

5. Shuo Zhang and Y. Bar-Shalom. Particle Filter Processing of Out-of-Sequence Measurements: Exact Bayesian Solution. pages 401 –404, dec. 2011.

6. Ananth Ranganathan, Michael Kaess, and Frank Dellaert. Fast 3D Pose Estimation with Out-of-Sequence Measurements. In *In IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2007.

7. M. Kaess, A. Ranganathan, and F. Dellaert. iSAM: Incremental Smoothing and Mapping. *IEEE Transactions on Robotics*, 24(6), 2008.

8. R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. g2o: A General Framework for Graph Optimization. In *Proc. of Intl. Conf. on Robotics and Automation (ICRA)*, 2011.

9. V. Indelman, S. Williams, M. Kaess, and F. Dellaert. Factor Graph Based Incremental Smoothing in Inertial Navigation Systems. In *Proc. of Intl. Conf. on Information Fusion (FUSION)*, 2012.

10. S. J. Julier and J. K. Uhlmann. Fusion of time delayed measurements with uncertain time delays. In *Proc. of American Control Conference*, 2005.

11. M. Choi, J. Choi, and W.K. Chung. State estimation with delayed measurements incorporating time-delay uncertainty. *Control Theory Applications, IET*, 6(15):2351–2361, 2012.

12. Jörg Fischer, Achim Hekler, and Uwe D. Hanebeck. State estimation in networked control systems. In *Proc. of Intl. Conf. on Information Fusion (FUSION)*, 2012.

13. J.-O. Nilsson, I. Skog, and P. Händel. Joint state and measurement time-delay estimation of nonlinear state space systems. In *Proc. of Intl. Conf. on Information Sciences Signal Processing and their Applications (ISSPA)*, 2010.

14. F.R. Kschischang, B.J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, February 2001.

15. GTSAM – The Georgia Tech Smoothing and Mapping Library. https://collab.cc.gatech.edu/borg/gtsam/.

16. Michael Kaess, Viorela Ila, Richard Roberts, and Frank Dellaert. The Bayes Tree: An Algorithmic Foundation for Probabilistic Robot Mapping. In *Proc. of Intl. Workshop on the Algorithmic Foundations of Robotics*, 2010.

17. Edwin Olson and Pratik Agarwal. Inference on Networks of Mixtures for Robust Robot Mapping. In *Proceedings of Robotics: Science and Systems (RSS)*, Sydney, Australia, July 2012.

18. Niko Sünderhauf and Peter Protzel. Switchable Constraints for Robust Pose Graph SLAM. In *Proc. of IEEE International Conference on Intelligent Robots and Systems (IROS)*, Vilamoura, Portugal, 2012.