Probabilistic Graphical Fusion of LiDAR, GPS, and 3D Building Maps for Urban UAV Navigation

Derek Chen and Grace Xingxin Gao

Abstract—Due to the recent interest in UAV commercialization, there is a great need for navigation algorithms that provide accurate and robust positioning in urban environments, which are often GPS challenged or denied. In this paper, we present a probabilistic graph-based navigation algorithm resilient to GPS errors. Fusing GPS Pseudorange and LiDAR odometry measurements with 3D building maps, we apply a batch estimation approach to generate a robust trajectory estimate and maps of the surrounding environment. We then leverage the maps to locate potential sources of GPS multipath and mitigate the effects of degraded pseudorange measurements on the trajectory estimate. We experimentally validate our results with flight tests conducted in GPS-challenged and GPS-denied environments.

Keywords—GPS, Urban Navigation, LiDAR, SLAM

I. INTRODUCTION

In recent years, Unmanned Air Vehicles (UAVs) have been sought after for numerous commercial applications [1], [2] including consumer package delivery, aerial photography [3], infrastructure inspection [4], [5] and emergency first response [6], [7]. In open-sky environments, Global Positioning System (GPS) receivers are most commonly used to provide accurate and globally referenced positioning for UAVs. However, many applications, such as consumer product delivery, require UAVs to operate in urban environments. In such environments, buildings and structures reflect and block GPS signals, causing multipath and low satellite visibility. These factors result in large positioning errors or causing GPS unavailable. Before widespread commercial adoption of these applications to occur, UAV navigation must be made safe and reliable.

In this paper, we present a sensor fusion algorithm based on a probabilistic graph model using GPS and LiDAR measurements with map data for urban UAV navigation. Our navigation algorithm is structured around odometry generated by Light Detection and Ranging (LiDAR) paired with compass measurements. To overcome drift and bias errors, GPS pseudorange measurements and LiDAR map-matching are applied to anchor the LiDAR odometry-derived trajectory to the global frame. As the UAV navigates, it maintains maps of its surrounding environment, initialized from 3D building data. With each LiDAR scan, the UAV updates its maps and optimizes for its trajectory states using a probabilistic graph approach. The updated environment map is then used to mitigate multipath and non-line-of-sight (NLOS) errors in the GPS measurements. This work is based on the Master thesis of the first author [8]. The main contributions of this paper are as follows:

1) A probabilistic graph approach to fuse GPS pseudorange and LiDAR odometry measurements aided by 3D building model maps for UAV trajectory estimation and mapping.
2) An evaluation of the algorithm for urban UAV navigation through a series of flight test in GPS-challenged and GPS-denied environments.

II. RELATED WORK

Approaches for mitigation of urban GPS measurement errors range from GPS sensor fusion to aiding GPS with information from environment maps.

A. GPS Sensor Fusion

GPS receivers are often paired with other navigation sensors to improve positioning accuracy and availability. One of the common sensors is inertial navigation system (INS) [9], [10]. GPS with INS aiding has been applied, along with other sensors, to multiple vehicular [11], [12] and UAV [13] applications and has been shown to provide position estimates where GPS is intermittently unavailable. Khagani et al. have demonstrated that fusing GPS/INS with a Vehicle Dynamic Model leads to major improvements in navigation accuracy in periods of GPS outages [14]. Klein et al. rely on a navigation solution and use pseudorange measurements as aiding to navigate periods of GPS outages. However GPS/INS is still subject to growing drift and biases in long periods of GPS outages or degraded GPS. Environment observing sensors such as the Light Detection and Ranging (LiDAR) sensor [15], [16], [17] have been used to generate odometry measurements by measuring changes in the observed environment. Dill et al. have shown that integrating LiDAR with INS is able to obtain globally anchor navigation solutions to meter-accuracy when GPS is available [18]. Soloviev has demonstrated that 2D line LiDAR scan matching can be paired with GPS for navigating in environments where GPS navigation alone is difficult [16]. More recently, these sensors have been combined in Simultaneous Localization and Mapping (SLAM) approaches [19], [20], [21], [22], [23], [24], [25], most commonly developed for indoor and GPS-denied navigation. While SLAM techniques alone have the capability of allowing users to navigating by observing environmental changes, long term positioning often suffers from drift and biases. By incorporating GPS, the positions and maps generated by the SLAM algorithms are anchored to the global frame [26]. Gao et al. implemented fusion of INS/GPS/LIDAR measurements for urban and indoor navigation using a hybrid scan matching algorithm [17]. Shepard et al. implemented a navigation system combining Parallel Tracking and Mapping (PTAM) processed camera measurements with carrier-phase differential GPS [26].
B. Improving Urban GPS with Environment Mapping

Recently, environment observation and modeling approaches have been used to mitigate multipath and NLOS errors in the GPS measurements and improve accuracy and availability in urban environments. Initials approaches applied upwards-facing omnidirectional cameras to characterize the city skyline surrounding a GPS receiver’s antenna to reject occluded signals [27], [28]. Zribi et al. combined GPS measurements, odometric data with a digital road map to improve 2D vehicle positioning [29]. With the recent availability of more accurate three-dimensional (3D) mapping, 3D building models have been leveraged to model environmental features that occlude or degrade GPS signal measurements [30]. Groves et al. applies shadow-matching [31], [32], a technique where 3D building frame, along with its GPS receiver clock biases.

A. UAV Trajectory Modeling

We define a UAV trajectory as a set of UAV states:

\[
X = \{x_0, x_1, x_2, \ldots, x_K\},
\]  

Each UAV state, \(x_k\), is denoted as the position and velocity of the UAV, in the Earth-centered Earth-fixed (ECEF) coordinate frame, along with its GPS receiver clock biases.

\[
x_k = \begin{bmatrix} x_k & y_k & z_k & \dot{x}_k & \dot{y}_k & \dot{z}_k & b_k & \dot{b}_k \end{bmatrix}^T
\]  

That is

\[
x_k = \begin{bmatrix} x_{k,\text{pos}} & x_{k,\text{vel}} & b_k & \dot{b}_k \end{bmatrix}^T,
\]

where

\(x_{k,\text{pos}}\) : position of UAV in ECEF

\(x_{k,\text{vel}}\) : velocity of UAV in ECEF

\(b_k\) : GPS Receiver Clock Bias

\(\dot{b}_k\) : GPS Receiver Clock Bias Drift Rate.

The GPS clock bias and clock drift are included in the UAV state vector to account for the difference between the receiver clock and the GPS satellite clocks. To propagate between sequential states, a constant velocity model describes the dynamics of the UAV. The state transition matrix of the UAV is defined as:

\[
H = \begin{bmatrix}
1 & 0 & 0 & dt & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & dt & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & dt & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & dt & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}.
\]

The state transition between UAV is modeled with the following equation:

\[
\begin{bmatrix}
x_{k,\text{pos}} \\
x_{k,\text{vel}} \\
b_k \\
\dot{b}_k
\end{bmatrix} = H
\begin{bmatrix}
x_{k-1,\text{pos}} \\
x_{k-1,\text{vel}} \\
b_{k-1} \\
\dot{b}_{k-1}
\end{bmatrix},
\]

where \(dt\) is the time increment between states.

B. LiDAR Odometry

We structure our UAV navigation solution on LiDAR odometry measurements. At each measurement epoch, the LiDAR generates a 3D point cloud of its surrounding environment in its own LiDAR frame. By comparing consecutively collected point clouds, and applying the Iterative Closest Point (ICP) algorithm [41], [42], we derive the relative motion of the UAV and generate these odometry measurements. Given a source and a reference point cloud, \(u\) and \(v\), the ICP algorithm seeks to find a translation and rotation transformation that would best match the two input clouds by minimizing the correspondence residual function:

\[
E(R, T) = \frac{1}{n_p} \sum_{i=1}^{n_p} ||u_i - Rv_i - T||^2,
\]

where

\(u_i\) : the \(i\)th point in 3D point cloud \(u\)

\(v_i\) : the \(i\)th point in 3D point cloud \(v\)

\(R\) : 3x3 rotation matrix

\(T\) : 3x1 translation vector

\(n_p\) : Number of corresponding points.

The steps in the ICP algorithm are as follows:

1) **Point Correspondence**

For each point in the source point cloud, the algorithm finds an associated point in the reference point cloud that is closest to the source point.

2) **Rejection**

Point correspondences that are either located too far or rotated too much are treated as outliers and are rejected.

3) **Minimize and Transform**

The residual function \(E(R, T)\) is then minimized to find the rotation and translation that best aligns the source and reference point clouds and applies the resulting translation and rotation to the source point cloud.
4) **Iterate**  
The process is iterated until convergence when the residuals are under a certain threshold or a maximum number of iterations is reached.

Figure 1 shows an example of two consecutive point clouds prior to the application of the ICP algorithm. Here we use the translation vector, \( T \), to define the relative change in position between two consecutive UAV states, \( k \) and \( k+1 \). The LiDAR odometry measurement vector \( \mathbf{l}_{k,k+1} \) is defined as:

\[
\mathbf{l}_{k,k+1} = \begin{bmatrix} l_{x,k,k+1} \\ l_{y,k,k+1} \\ l_{z,k,k+1} \end{bmatrix}^T,
\]

where each component of \( \mathbf{l}_{k,k+1} \) is situated in the local LiDAR frame. The relation between consecutive states,

\[
\begin{bmatrix} x_k \\ y_k \\ z_k \end{bmatrix} = \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ z_{k-1} \end{bmatrix} + \begin{bmatrix} l_{k,k-1,x} \\ l_{k,k-1,y} \\ l_{k,k-1,z} \end{bmatrix},
\]

is used to construct the LiDAR odometry measurement function:

\[
h_{\text{LiDAR}}(\mathbf{x}_k, \mathbf{x}_{k-1}) = \mathbf{x}_{k_{\text{pos}}} - \mathbf{x}_{k-1_{\text{pos}}} + \epsilon_{\text{ICP},k,k-1},
\]

where \( \epsilon_{\text{ICP},k,k-1} \) is the Gaussian error attributed to the ICP odometry.

1) **Compass Measurements**  
To integrate the LiDAR odometry with GPS, compass measurements provide heading information used to rotate the LiDAR measurements from the local LiDAR frame to ECEF. Assuming a level-UAV with a right handed coordinate frame (y-axis oriented forward-pointing and z-axis oriented upward-pointing), the LiDAR measurements are rotated into the East, North, and Up (ENU) coordinate frame with the following rotation matrix:

\[
\mathbf{x}_{\text{ENU}} = \begin{bmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{x}_{\text{Local}}.
\]

where \( \mathbf{x}_{\text{ENU}} \) is the position vector in the ENU frame, \( \theta \) is the rotation between the LiDAR from and north, and \( \mathbf{x}_{\text{Local}} \) is the position in the local LiDAR frame. With the latitude, \( \phi \), and longitude, \( \lambda \), of the UAV, the measurements are then rotated to ECEF with the following rotation matrix:

\[
\mathbf{x}_{\text{ECEF}} = \begin{bmatrix} -\sin(\lambda) & -\sin(\phi)\cos(\lambda) & \cos(\phi)\cos(\lambda) \\ \cos(\lambda) & -\sin(\phi)\sin(\lambda) & \cos(\phi)\sin(\lambda) \\ 0 & \cos(\phi) & \sin(\phi) \end{bmatrix} \mathbf{x}_{\text{ENU}}.
\]

Although estimation of the orientation of the UAV is beyond the scope of the paper and a fairly accurate rotation between the LiDAR and ENU frames are assumed, degradation in the compass measurements by environment effects may be detected by comparing the compass measurements with the rotations derived from the ICP algorithm. The incremental rotations are then used to estimate the orientation of the UAV and align the LiDAR frame with ENU. Furthermore, LiDAR urban map matching can be used to obtain reference locations matching the orientation of the UAV with the referenced structures in the surrounding environment. Any errors in frame alignment are absorbed as errors in the estimation step.

C. **LiDAR-based Map Matching**  
To reduce the effect of drift and bias from the LiDAR odometry measurements, the UAV incorporates LiDAR-based map matching. As the UAV navigates, the collected point cloud measurements are used to construct two types of maps to represent its environment, a point cloud map and an urban building map.

1) **Point Cloud Map Matching and Loop Closures**  
As the UAV travels, it concatenates all LiDAR point clouds collected throughout its trajectory and anchors them in their collecting positions in what is known as the point cloud map. When the UAV revisits these previously scanned areas, the point cloud map is then leveraged for loop closures. Through proximity, the feasibility of a loop closure is detected and initiated. The ICP algorithm is used to find the relative transformation between the current state and the closest prior state associated with the region in the point cloud map. To ensure accuracy of the loop closure, the association of the two states is only enacted when the residual function in the ICP algorithm is below a certain threshold, verifying that the two point clouds are matching scans of the same region and above a level of similarity. Once the association is verified, we form the ICP-derived translation vector relating the two states, \( \mathbf{l}_{i,j} = [l_{i,jx}, l_{i,jy}, l_{i,jz}] \),

\[
\begin{bmatrix} x_k \\ y_k \\ z_k \end{bmatrix} = \begin{bmatrix} x_j \\ y_j \\ z_j \end{bmatrix} + \begin{bmatrix} l_{k,jx} \\ l_{k,jy} \\ l_{k,jz} \end{bmatrix},
\]

The corresponding measurement function, \( h_{\text{Loop}}(\mathbf{x}_k, \mathbf{x}_j) \), is defined as:

\[
h_{\text{Loop}}(\mathbf{x}_k, \mathbf{x}_j) = \mathbf{x}_{k_{\text{pos}}} - \mathbf{x}_{j_{\text{pos}}} + \epsilon_{\text{ICP},i,j},
\]

where \( \epsilon_{\text{ICP},i,j} \) is the Gaussian error attributed to the ICP odometry.
2) Urban Map Matching

While navigating in areas where 3D building models are available, 3D building models are loaded and pre-processed into the urban building map to provide an alternative way to anchor the UAV positions to the global frame. Prior to urban navigation, the UAV first initializes the urban building map with the intended navigation area by pre-loading existing 3D building model data. In our experiments, we used the GPS-only navigation solution and pre-loaded building model data in a 1 km x 1 km square from the Champaign building footprint dataset [43]. The building footprint dataset consists of three-dimensional polygons referenced in WGS84 of buildings in Champaign, IL. Using the vertices of the polygons in the data, 3D models were generated consisting of 2D building shapes paired with corresponding building heights, shown in Figure 2. Navigating at low altitudes, the UAV extracts the outline from each building, shown in Figure 3, to be compared with the observations seen from each LiDAR point cloud. The set of points is defined as $M_{ubm}$. To identify comparable features between the urban building map and the LiDAR point clouds, LiDAR plane-fitting is used to extract walls and large recognizable features from the LiDAR point cloud. This is accomplished with the M-estimator SAmple Consensus (MSAC) algorithm [44], a variant of the RANdom SAmple Consensus (RANSAC) algorithm [45]. For each MSAC-generated plane, a set of points from the original point cloud is sampled and rotated to the corresponding frame of the urban building map. This process is shown in Figure 4 where the original point cloud is shown with the detected planes and points sampled from each plane. For epoch $k$, the set of sampled point measurements collected from all planes is defined as $Z_{s_k}$.

![Fig. 2. Urban building map of the Green Street area in Champaign, IL loaded from the Champaign Building Footprint data set. Each building consists of a building shape. Building height (not shown) is recorded in the system.](image)

![Fig. 3. Walls detected from the urban building map.](image)

![Fig. 4. Sampled points form planes detected in a point cloud.](image)

a) Particle-based Map Matching

Due to the varying geometry of the map that makes it difficult to optimize as a mathematical model, a Monte Carlo particle approach [46] is taken towards map matching. Particles are generated to represent the potential positions of the UAV, shown in Figure 5. We define the set of generated particles as set $M$. Any generated particles that intersect with buildings are regenerated.

For each UAV particle, $m_{k,i}$, the urban building map is used to generate a virtual range measurement to the closest structure in the direction of each measurement in $Z_{s_k}$. This is a process defined with the measurement function, $h_{ubm}(m_{k,i}, M_{ubm}, \phi_j)$ where $\phi_j$ is the angular direction of the $j$th measurement. The set of virtual range measurements via the urban building map is defined as set $Z_{ubm,i}$ for UAV particle $i$.

These virtual measurements are then compared to the observed measurement vector to generate a probability to determine the likelihood of a particle based on the observed measurements. For UAV particle $i$ modeling the urban map matching at epoch $k$, $m_{k,i} \in M$, the likelihood relation is formalized as:

$$P(m_{k,i}|Z_{s_k}) \propto \prod_j e^{-\frac{1}{2}||z_{s_k,j} - h_{ubm}(m_{k,i}, M_{ubm}, \phi_j)||^2},$$

where $z_{s_k,j}$ is the $j$th measurement in the set of sampled plane measurements collected at epoch $k$.

The particle with the highest likelihood that satisfies an empirical threshold is deemed the most likely position of the UAV. If no particle satisfies the threshold, then no urban
map matching measurement is made for that epoch. Figure 6 shows the result of the map matching between the initial estimate, the map matched estimate, and the true position along with the map-matching measurements. Once the most likely particle is identified, it is designated as an urban building map matching measurement defined as:

$$m_k = \begin{bmatrix} m_{k_x} & m_{k_y} & m_{k_z} \end{bmatrix}^T,$$

and its associated measurement function:

$$h_{Map}(x_k) = x_{k_{pos}} + \epsilon_{Map},$$

where

- $x_{k_{pos}}$: Position components of $x_k$
- $\epsilon_{Map}$: Map matching error term

D. GPS Measurements

To further anchor the UAV position in previously unmapped areas or where a good mapping fit could not be made, GPS pseudorange measurements are used. GPS pseudorange measurements measure the approximate distance between a GPS receiver’s antenna and the broadcasting GPS satellite. With at least four GPS pseudorange measurements, the UAV is able to obtain a position and time navigation solution. It is noted that although only GPS measurements are used in this paper, the similar approach would be applied to incorporate range measurements from other Global Navigation Satellite Systems (GNSS). The receiver clock bias and drift for each GNSS system would need to be included in the UAV state vector. The pseudorange, $\rho_k$, is modeled by the following measurement function:

$$h_{GPS}(x_k) = ||x_k^{(s)} - x_{k_{pos}}|| + b_k + \epsilon_{\rho_k},$$

where

- $x_k^{(s)}$: Position of GPS satellite $s$ at epoch $k$
- $x_{k_{pos}}$: Position components of UAV state $x_k$
- $b_k$: Clock bias of of UAV state $k$
- $\epsilon_{\rho_k}$: Pseudorange error term

Errors in the pseudorange measurements consist of the ionospheric, tropospheric, receiver noise, and environment multipath errors. When present and in severe cases, the most dominant errors result from environment multipath.

In this work, the reflected GPS signals that cause these errors are categorized as either NLOS signals or traditional multipath signals. NLOS signals occur when a GPS receiver receives a reflected signal from an satellite not-in-view, while multipath occurs when the GPS receiver receives a direct LOS signal to the satellite along with reflected signals from the environment.

For NLOS signals, an approach similar to [35], [47], [48] is taken. Line-of-Sight (LOS) vectors are formed between the GPS position estimate and each visible satellite. If the vectors intersect with buildings in the urban building map, then the measurement is characterized as affected by NLOS errors. We expand upon the previous approaches by using the LiDAR measurements to continuously update the building maps used for NLOS error detection. For multipath signals, due to the unpredictable nature of the angles of reflection that multipath can occur at, it is difficult to pinpoint their reflecting locations. As such, a de-weighting approach is applied for multipath mitigation. Using the LiDAR-updated urban building map, the proximity of buildings surrounding the navigation solution at each time step is incorporated to de-weigh the GPS components of the sensor fusion solution in the presence of numerous buildings. This allows the UAV to increase the weighting of the LiDAR odometry solution for navigation in multipath-affected regions.
IV. GRAPHICAL SENSOR FUSION

To estimate the UAV trajectory, we take a probabilistic approach to sensor fusion. We represent the UAV trajectory as a posterior probability conditional upon a UAV state prior and the set of all GPS, LiDAR odometry, and LiDAR map matching measurements,

\[
P(X|x_0, Z_{GPS}, Z_{LiDAR}, Z_{Map}, Z_{Loop}),
\]

where

\[
\begin{align*}
X & \quad : \text{UAV trajectory} \\
x_0 & \quad : \text{an initial prior on the UAV state} \\
Z_{GPS} & \quad : \text{the set of all GPS pseudorange measurements} \\
Z_{LiDAR} & \quad : \text{the set of all LiDAR odometry measurements} \\
Z_{Map} & \quad : \text{the set of all urban map matching measurements} \\
Z_{Loop} & \quad : \text{the set of all loop closure measurements.}
\end{align*}
\]

The initial prior of the UAV state is initialized with a GPS-derived navigation solution at the initial state and will be re-anchored once more reliable measurements are available.

A. Graphical Modeling

To model the posterior probability defined in Eqn(18), a Bayesian network [49], a type of directed and acyclic probabilistic graph, is applied. A Bayesian graph formulation has the following advantages:

1) Conditional dependencies between related states are applied as constraints in the estimation process, allowing for batch estimation. For example, this allows the presence of a GPS measurement at one node to anchor the rest of the graph to the global frame.

2) Batch estimation across the graph allows for update of past trajectory estimates, enabling generation of an optimized map.

3) Mapping and estimation are complementary. Incorporating map-based measurements improve the trajectory estimate while formulating an optimized trajectory estimate generates and updates an optimized map.

The graph is structured around two types of nodes, UAV nodes and anchor nodes. UAV nodes are the back bone of the Bayesian network and are used to represent the states of the UAV as random vectors throughout the trajectory. Anchor nodes represent visible GPS satellites and urban map matching points as landmarks in the environment and have an associated position vector. Directed edges then connect related nodes, representing relations formulated by a dynamic process or various types of sensor measurements. The backbone of the UAV trajectory graph is structured by the state transition and various types of sensor measurements. Directed edges then connect related nodes, representing relations formulated by a dynamic process or various types of sensor measurements. The graph to the global frame. Urban map matching nodes yield a similar effect but tie the respective UAV nodes to urban map matching anchors. Finally, loop closure edges relate non-consecutive UAV nodes through the LiDAR loop closure measurements.

Figure 7 shows an example of a UAV trajectory modeled by the described Bayesian network. For conciseness, GPS satellite nodes are grouped by each satellite PRN, yielding one node per satellite as opposed to one node per satellite per GPS measurement epoch.

The posterior probability of each UAV node is formulated from the probabilities of all connected directed edges to that node:

\[
P(x_k|x_{k-1}, z_k) \propto P(x_k|x_{k-1}) P(x_k|z_{GPS}) P(x_k|x_{k-1}, z_{LiDAR}) P(x_k|z_{Map}) P(x_k|z_{Loop}, x_j)
\]

where

\[
\begin{align*}
x_k & \quad : \text{UAV state at epoch } k \\
z_{GPS} & \quad : \text{GPS measurements at epoch } k \\
z_{LiDAR} & \quad : \text{LiDAR measurements at epoch } k \\
z_{Map} & \quad : \text{Map measurements at epoch } k \\
z_{Loop} & \quad : \text{Loop closure measurements at epoch } k
\end{align*}
\]

The posterior probability of the UAV trajectory, for all UAV states for each epoch from 1 to K, is then formulated from the joint probability of all individual posterior probabilities.

\[
P(X|x_0, Z) = P(x_0) P(x_1|z_1, x_0) P(x_2|z_2, x_1) \ldots P(x_K|z_K, x_{K-1})
\]

where \(P(x_0)\) is the probability of an initial prior, \(Z\) is the set of all sensor measurements and \(P(x_k|z_k, x_{k-1}, x_j)\) is the posterior probability of state \(i\) given measurements \(z_k\), the previous state \(x_{k-1}\), and any loop closure connected states \(x_j\).

B. Graphical Inference

To estimate the UAV trajectory from the posterior probabilities, graphical inference is applied across the graph to find the most likely configuration of UAV nodes that satisfy the constraints implied by the edges [49]. Taking a Gaussian assumption for the measurement and process models of the system, each probability is converted to a likelihood function from the corresponding measurement functions:

\[
P(x_k|p_k) \propto e^{-\frac{1}{2} \left| h_{GPS}(x_k, p_k^{(s)}) + h_k \right|^2 / \sigma_k^2} + \frac{1}{2} \left| h_{LiDAR}(x_k, x_{k-1}) - l_{k,k-1} \right|^2 / \sigma_{IC}^2 P_{k,k-1} \]

\[
P(x_k|x_{k-1}, l_{k,k-1}) \propto e^{-\frac{1}{2} \left| h_{LiDAR}(x_k, x_{k-1}) - l_{k,k-1} \right|^2 / \sigma_{IC}^2 P_{k,k-1}}
\]

\[
P(x_k|m_k) \propto e^{-\frac{1}{2} \left| h_{Map}(x_k, m_k) - m_k \right|^2 / \sigma_{Map}^2}
\]
\[ P(x_k|x_j, l_{k,j}) \propto e^{-\frac{1}{2}(h_{\text{Loop}}(x_k, x_j) - l_{k,j})^2/\sigma_{ICP,k,j}^2}, \quad (24) \]
\[ P(x_k|x_{k-1}) \propto e^{-\frac{1}{2}(f(x_{k-1}) - x_k)^2/\sigma_{\text{Process}}^2}, \quad (25) \]

where \( f(x_{k-1}) \) is the state transition function, \( \sigma_{\rho_k}^2 \) is the variance of the pseudorange measurements, \( \sigma_{ICP,k-1}^2 \) is the variance of the LiDAR ICP measurements, \( \sigma_{Map}^2 \) is the variance of the map matching measurements, and \( \sigma_{\text{Process}}^2 \) is the process noise.

This is accomplished through Maximum a Posteriori (MAP) estimation. The negative log-likelihood is then taken of the exponential likelihood functions to formulate a non-linear least-squares optimization problem:

\[ \hat{X} = \arg \max_X P(X|Z, U) \quad (26) \]
\[ = \arg \min_X \sum \log P(X|Z, U) \quad (27) \]

\[ = \arg \min_X \sum \log \left| h_{\text{GPS}}(x_k, \hat{x}_k^{(s)}) - \rho_k^{(s)} \right|^2_{\sigma_k^{(s)}} + \sum \left| h_{\text{LiDAR}}(x_k, x_{k-1}) - l_{k-1} \right|^2_{\sigma_{ICP,k-1}^2} + \sum \left| h_{\text{Map}}(x_k) - m_k \right|^2_{\sigma_{Map}^2} + \sum \left| h_{\text{Loop}}(x_k, x_j) - l_{k,j} \right|^2_{\sigma_{ICP,k-j}^2} + \sum \left| f(x_{k-1}) - x_k \right|^2_{\sigma_{\text{Process}}^2} \quad (28) \]

where

\[ \hat{X} \quad \text{: Estimated UAV Parameters} \]
\[ \sum \quad \text{: Summing across all measurements} \]
\[ \sum \quad \text{: Summing across all epochs} \]
\[ \text{||} \cdot \text{||}_2^2 \quad \text{: Squared Mahalanobis distance with covariance} \quad \sigma^2 \]

\[ \Delta x_i(x_i, z_i) = -\text{pinv} \left( J_i^T W_i J_i \right) J_i^T W_i e_i(x_i, z_i), \quad (29) \]

where \text{pinv} is the pseudo inverse operation and the Jacobian, weighting matrix, and error function at time step \( i \) are defined as \( J_i, W_i, e_i \), respectively.

The Jacobian for the state transition process model is based on the state transition matrix \( H \) defined in Eqn. 4.

\[
J_{\text{Process}} = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}. \quad (30)
\]

\[
J_{GSPk} = \begin{bmatrix}
\frac{x(1) - z_k}{||x(1) - z_k||} & \frac{y(1) - y_k}{||y(1) - y_k||} & \frac{z(1) - z_k}{||z(1) - z_k||}
\end{bmatrix} \quad (31)
\]

The Jacobian for the ICP-derived LiDAR odometry, described in Eqn. 9, relates the two consecutive states connected by the LiDAR odometry edge:

\[
J_{\text{LiDAR}} = \begin{bmatrix}
I_3 & 0_{6 \times 6}
\end{bmatrix}. \quad (32)
\]

The Jacobian for the ICP-derived LiDAR loop closure, described in Eqn. 13, relates two non-consecutive states:

\[
J_{\text{Loop}} = \begin{bmatrix}
I_3 & 0_{6 \times 6}
\end{bmatrix}. \quad (33)
\]

Similarly, the ICP-derived LiDAR loop closure, described in Eqn. 13, relates two non-consecutive states:

\[
J_{\text{Map}} = \begin{bmatrix}
I_3 & 0_{6 \times 6}
\end{bmatrix}. \quad (34)
\]

For \( M \) sensor measurements, the trajectory Jacobian, \( J_{\text{traj}} \), is a \( M \times 8K \) matrix formulated by stacking the the Jacobian from each sensor measurement into a large but sparse matrix. A 3x3 identity matrix is included for the initial prior of the
UAV state. For conciseness $J_P = J_{\text{Process}}$, $J_G = J_{\text{GPS}}$, $J_L = J_{\text{LiDAR}}$, $J_M = J_{\text{Map}}$, and $J_O = J_{\text{Loop}}$. The weighting matrix of the trajectory $W_{\text{trajectory}}$ is defined as a diagonal matrix:

$$W_{\text{trajectory}} = \begin{bmatrix} W_{\text{Prior}} & W_{\text{GPS}} & W_{\text{LiDAR}} & W_{\text{Map}} & W_{\text{Loop}} \end{bmatrix}$$

(42)

where $W_{\text{Prior}}$, $W_{\text{GPS}}$, $W_{\text{LiDAR}}$, $W_{\text{Map}}$, $W_{\text{Loop}}$ are the block diagonalized weight matrices for the Prior and all GPS, LiDAR, Map, and Loop closure measurements respectively.

Iterative least squares is applied to the Jacobian and error vector to estimate the trajectory parameters until convergence or when a maximum number of iterations are reached in batch estimation iterations. Although computation speed is not addressed in this paper, efficient optimization methodologies for solving Bayesian graphs [51], [52] can be applied to solve for the UAV trajectory parameters in near real-time. Only when significant changes to the graph occur does a large batch estimation need to be computed. As such, computationally-manageable incremental updates are made at each time step while the large scale batch processing segments can be offloaded to cloud-based computing. After computation, the large scale batch estimate is then incorporated to update the on-board graph.

C. Map Update

After trajectory estimation, the UAV’s environment maps are updated. To update the point cloud map, the point clouds are re-anchored at the updated UAV node positions. For the urban building map, the points on the urban building map can be updated using globally referenced building data points, such as the Champaign building footprint data set [43] and generated through plane-fitting of the LiDAR point clouds. Since the building points are globally referenced, those points remain as a constant source of positioning. The sampled plane points are added to map new structures and correct existing building points. As a constant source of positioning. The sampled plane points are added to map new structures and correct existing building points. To reduce the possibility of adding dynamic objects, such as vehicles and pedestrians, to the map, the LiDAR data is pre-processed for points that are either too distant or too close to the LiDAR sensor. Any remaining objects mapped either are filtered out during the plane fitting process or are minimal enough in the total point cloud to have a negligible effect on the map-matching solutions. For future work, object recognition techniques such as those described in [53], [54] may be applied to identify and remove non-structural objects.

D. Graphical Sensor Fusion Summary

Figure 8 shows a summary of the graphical sensor fusion process. First, the graph is initialized with an initial guess of its location from available GPS and map-matching measurements. As the UAV navigates, it collects measurements from available sensors and updates the graph with new nodes and constraining edges. Incremental graphical inference steps are then performed to estimate its most recent states and respective components of the map are updated. Batches of
measurements are also collected by an offloading processor which periodically performs inference on larger segments of the trajectory, returning an optimized trajectory and updated urban building map and point cloud map.

![Fig. 8. System Flow Diagram for Graphical Sensor Fusion](image)

V. RESULTS AND DISCUSSION

A. Experimental Setup

To validate the proposed algorithm, data was collected and flight tested on a UAV test bed. The flight test was conducted on the iBQR, shown in Figure 9, a custom built quadrotor in the research group [55]. The iBQR was designed with configurable arms and sensor mounts so that GPS signals can be received without interference from the UAV’s flight electronics. For both testbeds, the same data collection platform was used. Data is collected on the Asctec Mastermind, an Intel i7 computer operating in Ubuntu Linux and running the Robotics Operating System (ROS). The Asctec Mastermind connects to the compass, GPS receiver, and LiDAR. For our heading measurements, we use magnetometer of the Xsens MTi IMU, an integrated inertial measurement unit. GPS pseudoranges are collected from commercial off-the-shelf (COTS) u-blox LEA-6T GPS receiver paired with a Maxtena helical antenna. Finally, we mount a Velodyne Puck Lite LiDAR for LiDAR point cloud collection. In both cases, navigation data is collected for post-processing.

![Fig. 9. iBQR Data Collection Platform](image)

B. Results and Analysis

Dynamic flight data was collected on a tethered iBQR flight where the UAV travelled from a GPS-friendly environment, to a GPS-challenged environment, and finally to a GPS-denied scenario. The GPS conditions for each region of the selected trajectory are highlighted in Figure 10. In the GPS-friendly environment, an average of nine satellites were visible, leading to an accurate navigation solutions. As the UAV travels into the GPS-challenged region, although a similar number of satellites are visible, multipath and NLOS effects degrade the quality of the GPS measurements and the GPS navigation solution. Finally, in the GPS-denied region, the UAV loses GPS satellite visibility due to the tall structures in the surrounding environment. Figure 11 shows the number of visible GPS satellites, NLOS satellites detected by the urban building map, and occluded satellites throughout the trajectory. Since it is challenging to obtain truth for outdoor UAV positioning, the pilot-observed trajectory aided by a georeferenced map of the area was used to estimate the true trajectory for reference. In Figure 12, we first show that the navigation solution of graphical sensor fusion with only the GPS pseudoranges with LiDAR odometry. Although a navigation solution is available throughout the UAV’s trajectory, severe multipath degrades the navigation solution and loss of GPS results in drift. Through the incorporation map matching and multipath mitigation with the urban building map, Figure 13 shows that the proposed augmented graphical sensor fusion approach is able to provide a robust and accurate trajectory estimate for the UAV in GPS-challenged and GPS-denied urban environments. The importance of an accurate trajectory estimate for mapping also becomes apparent after observing the point cloud map. Figures 14 and 15 show the point cloud map generated by anchoring the point clouds to the UAV trajectory positions. With a GPS-only Kalman position output, once the UAV entered the GPS-challenged area, mapping of the region becomes noisy, as seen in Figure 14. With the graphical approach where the trajectory is batch estimated and the noisy pseudorange measurements are mitigated, Figure 15 shows a more accurate
Fig. 11. Number of visible GPS satellites, NLOS satellites detected by the urban building map, and occluded satellites.

Fig. 12. Experimental results from LiDAR odometry + GPS graphical sensor fusion in an urban environment. Pairing LiDAR odometry measurements with GPS pseudoranges provides an improved navigation solution than Kalman Filter GPS alone, however subsections of multipath degrade the navigation solution.

Mapping of the environment. Finally, we examine the effects of the loop closure measurement on the trajectory estimate. Data were collected as the UA V navigated into and out of an urban canyon forming alleyway. The results of our navigation algorithm is compared with the COTS receiver output and trajectory estimate without loop closure in Figure 16. Since the alleyway creates a difficult GPS-challenged environment, the Kalman filtered GPS navigation solution is degraded due to low satellite visibility and multipath. With the uniformity of the LiDAR observable features (i.e. straight walls) in the alleyway, the ICP algorithm fails to generate an accurate odometry estimate. Without loop closure, the trajectory estimates degrade with drift and both the GPS only and GPS with LiDAR positions deviate from the true path. Application of loop closure allows the UAV to constrain its motion at the start and end of the trajectory, leading to a more accurate trajectory estimate in the alleyway.

Fig. 13. Experimental results from full probabilistic graphical algorithm in an urban environment. With the map-aiding in the form of LiDAR map matching and multipath and NLOS mitigation, we obtain an improved trajectory estimate over a GPS-only Kalman filtered approach.

Fig. 14. Point cloud map (in the LiDAR frame) generated from GPS-only navigation solution. In the GPS-challenged and GPS-denied regions, the point cloud map becomes noisy and mis-positioned.

VI. Conclusion

In this paper, we presented a graphical sensor fusion approach for urban UAV navigation. In our approach, we applied a probabilistic graph to fuse GPS, LiDAR, and compass measurements with urban building footprint data. First, we used the LiDAR point cloud measurements in three ways: LiDAR odometry and LiDAR-based map matching (point cloud matching and urban map matching). Using the ICP algorithm on consecutive LiDAR scans, we generate odometry measurements to determine the relative motion of the UAV. Then, we applied plane-fitting on the LiDAR point clouds to detect buildings walls for map matching with our urban building map. To improve our GPS measurements, we then incorporated techniques for multipath mitigation. Using the environment map generated by the LiDAR, and aided by the
In urban building footprints, we identified the presence of multipath and NLOS errors from the environment surrounding a UAV and mitigated their effects on the GPS pseudorange measurements. Combining these elements, we then presented an algorithm for UAV trajectory estimation and environment mapping. Through a series of experiments in GPS-challenged environments, we have shown that our graph-based approach has the capability of combining the advantages of the LiDAR and GPS sensors. This enables our UAV to obtain an accurate trajectory estimate in situations that are normally challenging when navigating with each of the sensors individually.

ACKNOWLEDGMENT

The authors would like to thank Akshay Shetty, Shubhendra Chauhan, Craig Babiarz, Arthur Chu, Sri Ramya Bhamidipati, and Anneliese Fang for their assistance during numerous experiments and flight tests, and Enyu Luo for his hardware expertise in developing and supporting the operation of the iBQR.

REFERENCES


Derek Chen is currently a Masters student in the Department of Aerospace Engineering at the University of Illinois at Urbana-Champaign. He graduated with his B.S. degree in Aerospace Engineering from the University of Illinois at Urbana-Champaign in May 2014. His research interests are in autonomous systems and sensor fusion.

Grace Xingxin Gao received her B.S. degree in mechanical engineering and her M.S. degree in electrical engineering from Tsinghua University, Beijing, China in 2001 and 2003, respectively. She received her PhD degree in electrical engineering from Stanford University in 2008. From 2008 to 2012, she was a research associate at Stanford University. Since 2012, she has been an assistant professor in the Aerospace Engineering Department at University of Illinois at Urbana-Champaign. Her research interests are systems, signals, control, and robotics. She is a senior member of IEEE and a member of ION.