Long-range GPS-denied Aerial Inertial Navigation with LIDAR Localization

Garrett Hemann, Sanjiv Singh, and Michael Kaess

***Abstract*— Despite significant progress in GPS-denied autonomous flight, long-distance traversals (> 100 km) in the absence of GPS remain elusive. This paper demonstrates a method capable of accurately estimating the aircraft state over a 218 km flight with a final position error of 27 m, 0.012% of the distance traveled. Our technique efficiently captures the full state dynamics of the air vehicle with semi-intermittent global corrections using LIDAR measurements matched against an a priori Digital Elevation Model (DEM). Using an error-state Kalman filter with IMU bias estimation, we are able to maintain a high-certainty state estimate, reducing the computation time to search over a global elevation map. A sub region of the DEM is scanned with the latest LIDAR projection providing a correlation map of landscape symmetry. The optimal position is extracted from the correlation map to produce a position correction that is applied to the state estimate in the filter. This method provides a GPS-denied state estimate for long range drift-free navigation. We demonstrate this method on two flight data sets from a full-sized helicopter, showing significantly longer flight distances over the current state of the art.**

I. Introduction

Autonomous and manned aerial vehicles rely on accurate localization for safe navigation and for finding their destination. Relying on only inertial sensors for localization is not feasible because navigation drift accumulates over time without bounds. Typically, aerial vehicles complement inertial sensing with a global positioning system such as GPS to achieve drift-free navigation. While this is a simple solution to integrate, GPS is not always a reliable sensing mechanism. For one, satellite coverage breaks down from obstructions and multipath caused by mountains or skyscrapers. Also, GPS is susceptible to adversarial jamming and spoofing, rendering it useless or outright dangerous for navigation. Additionally, the satellite system is not immune against system errors, such as the January 2016 incident where wrong time information was temporarily broadcast after the decommissioning of a GPS satellite. Lastly, GPS only provides a navigation solution on Earth and extra-terrestrial navigation requires a different sensing strategy.

GPS-denied navigation has gained much attention within the past decade, particularly in the indoor and underwater domains. The problem is challenging because in the absence of a global reference such as GPS, onboard sensors can only produce a drifting navigation solution with unbounded error. Strategies for remaining localized with onboard sensors involve using a priori maps (e.g. [5]) as well as using simultaneous localization and mapping (SLAM) to build maps on the fly (e.g. [3]). Development in perception and efficient mapping algorithms have merged to form stable visual-based localization methods using feature-rich environments for GPS-denied navigation. These methods require repetitive landmark observations for loop-closure to eliminate longer term drift errors. This dependency, along with the payload limitations of small indoor air vehicles, limits the range with which localization algorithms can be demonstrated.

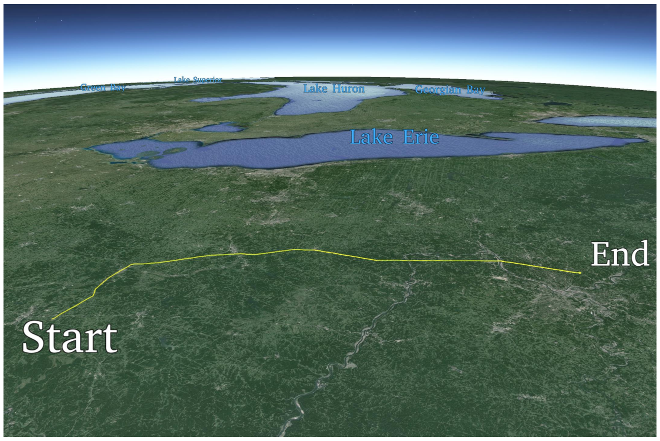

Fig. 1: GPS-denied 218 km helicopter flight (yellow trajectory), with a final position error of 27 m, 0.012% of the distance traveled. The helicopter took off from Zanesville airport, OH, and landed at Cedar Run airport, PA.

One area that has received less attention is extending the GPS-denied navigation capabilities to long distance outdoor missions (see Fig. 1). This scenario presents two major challenges: (1) capture the dynamics of the vehicle at a high rate and (2) estimate its 6 degrees of freedom (DOF) global position and orientation in real-time. The first challenge has been addressed for air vehicles in shorter range missions using Kalman filters or smoothing techniques (e.g. [4]). The second challenge however has not been addressed for long distances without GPS, and we present our strategy here.

To achieve drift-free localization in the absence of GPS, we match light detection and ranging (LIDAR) measurements against a digital elevation model (DEM) for localization. Our technique fits into the category of terrain referenced navigation (TRN) [8] in which terrain models are used for localization. The a priori DEM provides a surface elevation map at various resolutions for the entire desired flight plan. Instead of trying to scan a single LIDAR projection against the entire geographical region or world, we use our inertial propagation estimate to narrow the search space, reduce computational cost, and provide real-time global localization.

This work was funded by the Office of Naval Research (prime award N00014-12-C-0671) through Near Earth Autonomy Inc.

The authors are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA. {ghemann, ssingh, kaess}@cmu.edu

The low-drift estimate also accounts for periods of inactive corrections which may occur when flying at low altitudes, over water where LIDAR returns fail, dense fog, or in areas with low terrain variability.

Our work makes the following contributions:

1) We present a LIDAR localization algorithm that eliminates the need for cameras and therefore works independent of lighting conditions.
2) We provide a tightly-coupled LIDAR-inertial integration that achieves low, bounded position and orientation error using intermittent position corrections in the absence of GPS.
3) We demonstrate our ability for long distance drift-free navigation on two datasets from the flight of a full-sized helicopter, each covering around 200 km from takeoff to landing, a significant increase in distance over the current state of the art.

II. Related Work

Research in GPS-denied navigation on smaller unmanned aerial vehicles (UAVs) has received much attention in the past decade. The ubiquity of cheap, easily available platforms like quadrotors have opened opportunities for many different areas of autonomous aerial research. Unfortunately, smaller platforms are payload limited and restrict the range at which long distance autonomy can be tested. Therefore, not as much attention has been given to long distance aerial navigation.

Some visual odometry (VO) techniques for indoor GPS-denied autonomy have been extended to attempt longer range missions. Weiss et al. [11] discusses the state of the art for minimal payload aerial vehicles using a monocular-camera and inertial measurement unit (IMU) sensor setup. Their parallel tracking and mapping (PTAM) based visual SLAM approach was demonstrated on a 350 m flight with a final position error of only 1.47 m, at which point the battery was depleted. Another long-range visual odometry technique by Warren et al. [10] uses a deformable stereo-rig baseline to account for the vibrations of the vehicle and improve depth accuracy. This was tested on a fixed-wing UAV on a 6.5 km dataset. Currently the longest VO demonstration we know of is Zhang and Singh [12]. This work reduces translational drift of an inertial navigation solution by reparametrizing features of a downward-facing camera along a ground plane normal, extracted from a laser altimeter. On trajectories over 30 km they achieved an impressive 0.09% trajectory error. While adding VO improves on inertial-only navigation solutions, localization error still increases with traveled distance without bounds.

To address the issue of long-range drift from visual or inertial solutions, a ground referencing strategy is required. Quist [7] uses radar odometry to estimate the position of artificial ground radar scatterers. This process uses a Hough transform on a radar signal to identify the targets and their relative distance. Flight tests of their radar system with a commercial-grade IMU show comparable results in drift error to using a navigation-grade IMU only. Their flight covered 2.4 km with a final drift of 2.3% over the distance flown. The strength of using radar in this method gives the system the ability to operate in more varied weather conditions than a LIDAR, however it depends on artificial radar scatterers to be placed in the environment, which is often not a feasible solution.

A more effective ground referencing strategy is terrain referenced navigation because it uses previously generated elevation models of the natural terrain shape and does not require placing artificial markers in the environment. LIDAR scans of a landing strip are used by de Haag et al. [1] to navigate a 300 s landing sequence. The LIDAR points are converted to elevation estimates and aligned with the DEM using a sum of squared error minimum. While the landing drift error was considerably low (less than a meter in any direction), the system incorporated GPS and radar altimeters and only used the LIDAR when the data was reliable.

A more recent TRN-based method by Johnson and Ivanov [5] evaluates the horizontal position accuracy of a lunar lander approach by converting a LIDAR scan to an elevation model. They perform the matching using the Fast Template matching algorithm by Lewis [6] and finding the correction from the shift of the maximum value in the correlation map. We also use a normalized cross-correlation matching method, but ours does not rely on precomputed integral tables. This TRN approach was tested on Earth terrain models using 3 m DEM and their landing trajectory guaranteed 90 m accuracy. Their system however does not have a tightly coupled navigation solution. We extend this work to incorporate a high-rate fused state estimate, global corrections in three axes instead of two, and operate over long distance flights as opposed to just landing sequences.

The global correction method is just one half of the state estimation problem. Highly dynamic platforms like UAVs also require a continuous high-rate state estimate, typically at a higher resolution than what the global corrector provides. Most modern multi-sensor robotic platforms rely on efficient sensor fusion algorithms to capture a consistent state estimate from various input sensor rates. A common method is the Kalman filter which handles propagating uncertainty and providing a fused state estimate. A modified version of the Kalman filter, or error-state Kalman filter, has been used extensively by Trawny et al. [9] for state estimation in planetary landing. The state is propagated with IMU input and the Kalman filter estimates the time-varying IMU biases. They use a camera to track features from craters and correct the state estimate. We use a similar state estimation technique with the modified EKF, but incorporate a different correction method based on LIDAR.

Previous approaches to the long-distance GPS-denied state estimation challenge have used various types of sensing modalities and filter techniques to reduce track error and total trajectory error. However, the limitations in the operational environment or platform have prevented testing the long distance robustness of these algorithms. Furthermore, most of the techniques have only utilized either a robust filter for IMU bias estimation or an efficient terrain matching algorithm. Here we present a fusion of some of these techniques and improve them individually for considerable improvement in terrain matching and pushing the state of the art in long-distance GPS-denied navigation.

III. LIDAR Localization

Our LIDAR localization intermittently localizes the helicopter by aligning LIDAR measurements to a geo-referenced Digital Elevation Model (DEM). Our goal is to estimate our true 3D world position by searching for an optimal alignment between a 3D LIDAR point cloud as measured by the helicopter and the a priori DEM. We make use of the current position estimate to restrict the search area to a local DEM neighborhood. This requires a low-drift estimate, which is achieved by tightly coupled LIDAR-inertial fusion as discussed in section IV.

We decompose the 3D localization problem into a 2D translation and a 1D altitude problem that are solved sequentially. The DEM is given as a regular grid of elevation values, represented as a floating-point valued image $\mathbf{I}_D$. The point cloud from the helicopter LIDAR is converted into an image $\mathbf{I}_L$ of the same grid size by binning, allowing for noise filtering in the process. The localization problem is thereby reduced to finding the 2D offset between both images that provides the best correlation between DEM and LIDAR data. Finally, to obtain a full 3D position correction, the difference in elevation between predicted and measured ground surface is estimated. Only translation is estimated with this process since small angle misalignments cannot be recovered from local LIDAR data. Instead, orientation is corrected in the filter (see section IV).

A. LIDAR Binning

LIDAR binning takes the original LIDAR measurements and transforms them into a virtual DEM. The LIDAR measurements are obtained from a downward-facing 2D line LIDAR sensor rigidly attached to the vehicle. The individual measurements ${}^{L}\mathbf{p}_i$ of each scan are transformed from the LIDAR frame into the global frame using a time stamped state estimate of the body-to-world transform ${}^{G}_{V}\mathbf{R}(t)$ and a rigid transform of the sensor relative to the vehicle ${}^{V}_{L}\mathbf{R}$, using the transformation chain

$$ {}^{G}\mathbf{p}_i = {}^{G}_{V}\mathbf{R}(t) \cdot {}^{V}_{L}\mathbf{R} \cdot {}^{L}\,\mathbf{p}_i, \tag{1}$$

where G denotes the global frame, V the vehicle frame and L the LIDAR frame. To cover a region of sufficient size for reliable offset estimation, a sufficient number of line scans are accumulated, depending on the vehicle's velocity.

A robust heightmap is computed by grouping the 3D points into bins along the x-y plane at the exact resolution of the DEM. These bins contain an array of estimated ground elevation values predicted from the LIDAR returns. Bins with a small range show an area with high certainty that the LIDAR beams reflected off of the true ground surface. Bins with high variability on the other hand indicate occlusions such as trees, with only some beams reflected off the desired ground surface. Fig. 2 shows an example of point bins that have as much as 30 m in range variation due to foliage. The lowest elevation per bin (marked red) is the closest representation of the true ground elevation for that cell. To improve the likelihood that each bin has at least one return from the ground, we discard bins with less than 30 points (we get up to 500 points per bin at cruising altitude). The result is a robust 2D heightmap $\mathbf{I}_L$ based on the lowest valued point of each valid bin. We have found this approach to generate similar height maps despite vegetation variations across different seasons.

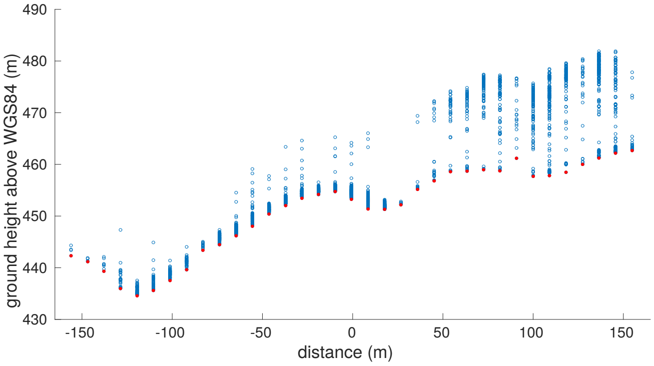

Fig. 2: A cross section of 3D LIDAR points (blue circles) after binning. The right half of the bins show high variability in height due to the presence of trees. The lowest point in each bin (marked in red) shows the smooth surface of the actual terrain.

Finally, the height map is smoothed using a mask. A mask is required because removing invalid bins creates holes in the image. A binary mask $\mathbf{I}_v$ is used to track the active valid cells. Active regions are smoothed with a Gaussian kernel. This produces $\mathbf{I}_L$ to be a smooth surface-like heightmap generated from the 3D LIDAR point cloud. Examples are shown in column (c) of Fig. 3.

B. LIDAR DEM Matching

With both measured and ground truth heightmap available, the next step is to identify an offset based on the best alignment. The DEM image $\mathbf{I}_D$ is created from a local neighborhood window around the current state estimate. To account for the uncertainty of the state estimate, a region larger than the LIDAR image $\mathbf{I}_L$ is searched. We perform an exhaustive sliding window search over offsets between $\mathbf{I}_L$ along $\mathbf{I}_D$, calculating a normalized cross-correlation value

$$ \mathbf{NCC}_{(i,j)} = \frac{1}{N(\mathbf{I}_v)} \frac{\sum(\mathbf{I}_{D,(i,j)} - \mu_{D,(i,j)})(\mathbf{I}_L - \mu_L)}{\sigma_D \sigma_L} \tag{2}$$

at each offset (i, j) to obtain a cross-correlation map (see column (d) of Fig. 3 for examples). Note that the height images are multiplied element-wise. Also note that the summation is over all valid pixels of the image, where $N(\mathbf{I}_v)$ is the number of valid pixels in image mask $\mathbf{I}_v$.

Given this correlation map, we can now extract the position offset between the prior state estimate and the true position. The maximum value of this normalized cross-correlation cost map represents the highest correlation match between the LIDAR image $\mathbf{I}_L$ and a sub region of the DEM $\mathbf{I}_D$ at index (i^*,j^*). The shift from the center of $\mathbf{I}_D$ to

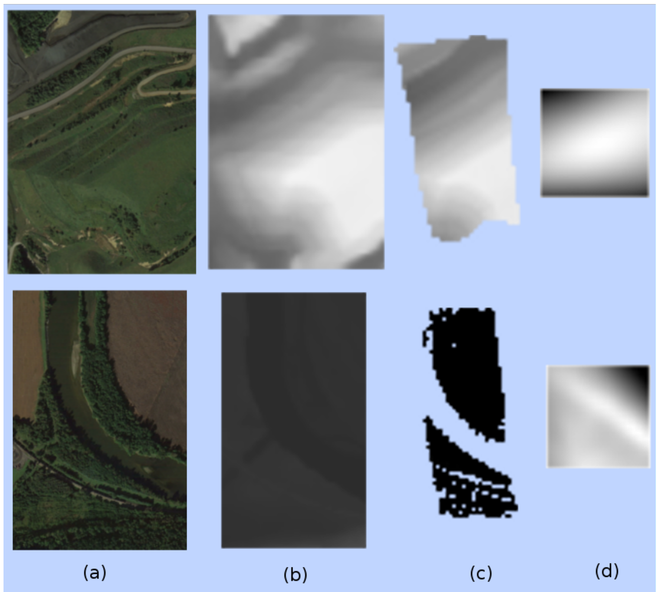

Fig. 3: Cost map of a successful match (top row) and an unsuccessful match (bottom row). The columns show (a) an aerial image of the terrain, (b) a DEM region $\mathbf{I}_D$ of the same area at 10m resolution, (c) the LIDAR projection $\mathbf{I}_L$ after binning and filtering, and (d) the normalized cross-correlation image of the matching (white denotes higher correlation). The successful match (top) has an $\mathbf{NCC}$ image with a distinct optimum near its center, whereas the failed match (bottom) shows no clear optimum because of the low variation in terrain elevation.

(i^*,j^*) represents the horizontal correction to the estimated vehicle position. If max$(\mathbf{NCC})$ is greater than an empirically determined threshold (we use 0.922), then it is a confident match and is used to correct the position. Otherwise we treat it as a failed match and no correction is sent to the filter.

For robustness, we also require some level of variability in the terrain before we accept a match. The matching algorithm only matches similarities in the images and ignores the actual shape and variation of the terrain. To ensure the terrain provides sufficient constraints, the standard deviation of all valid elevation values in the LIDAR image is computed. A successful match is returned only if this standard deviation is above a threshold (we use 2.5 m).

Two examples of LIDAR-DEM matches are shown in Fig. 3. The top row shows a successful match with terrain variability in multiple directions, giving a single optimal location for the matching. The bottom row shows a failed match caused by nearly flat terrain within the matching region, seen by the low contrast of $\mathbf{I}_L$ and $\mathbf{I}_D$, and due to a river running through the center that is difficult to detect for the LIDAR, leading to missing data in $\mathbf{I}_L$.

C. Altitude Correction

The final step is to compute a vertical correction (z) from the elevation images. The DEM image at the optimal index of the lateral matching $\mathbf{I}_{D,(i*,j*)}$ is subtracted element-wise from the measured heightmap $\mathbf{I}_L$. The mean of the resulting difference image is the average elevation offset between the estimated and the true altitude. The offset represents the state estimation error as measured by the LIDAR against the ground truth DEM.

IV. State Estimation

Robust state estimation relies on inertial sensing to be paired with an additional sensor to eliminate drift. Given an inertial sensor with a zero-mean Gaussian noise model, a Kalman filter can be used to fuse these different sensing modalities to obtain a drift-free high-dynamic state estimate. The extended Kalman filter (EKF) is typically used to model nonlinear systems, but it suffers from linearization errors. Instead of modeling the full state, a simplification of the algorithm is to propagate the error state. This type of filter, known as an error-state Kalman filter, improves the accuracy while still correctly modeling the system dynamics. Inertial navigation will then be sufficiently accurate to provide a robust state estimate between the LIDAR localization measurements.

The state we are estimating consists of vehicle orientation, velocity, position and IMU (accelerometer and gyroscope) biases. This is represented within a 16-dimensonal vector as

$$\begin{aligned}&\mathbf{x}(t) = \\ &\begin{bmatrix} {}^G_L\mathbf{q}^\top(t) & {}^L\mathbf{b_g}^\top(t) & {}^G\mathbf{v}^\top(t) & {}^L\mathbf{b_a}^\top(t) & {}^G\mathbf{p}^\top(t) \end{bmatrix}^\top\end{aligned} \tag{3}$$

where our notation closely follows [9]. We represent the orientation of the vehicle as a rotation of the global frame {G} with respect to the local frame {L} in the form of the quaternion ${}^G_L\mathbf{q}(t)$. We denote the equivalent 3x3 rotation matrix as $\mathbf{C_q}$. Therefore, our rotation representation transforms a vector from the local to the global frame as ${}^G\mathbf{v} = \mathbf{C_q} \cdot {}^L\mathbf{v}$. The gyroscope bias ${}^L\mathbf{b_g}^\top(t)$nd accelerometer bias ${}^L\mathbf{b_a}^\top(t)$ are represented in the local frame. The vehicle velocity ${}^G\mathbf{v}^\top(t)$ and position ${}^G\mathbf{p}^\top(t)$ are expressed in the global frame. The global frame is an earth-centered, earth-fixed (ECEF) reference frame that remains Cartesian even over long distance sprints and is not corrupted by earth's curvature. This also simplifies the model of earth's rotation, taken as ${}^G\boldsymbol{\omega}_\mathbf{e}$. The global gravity vector ${}^G\mathbf{g}$ is calculated at each time step using the WGS84 reference ellipsoid as described in Farrell [2].

A. State Propagation

The vehicle state is first estimated by propagating the kinematic equations given the input IMU sensor data. The measured IMU data, angular velocity ${}^L\boldsymbol{\omega}_m(t)$ and linear acceleration ${}^L\mathbf{a}_m(t)$, is modeled as

$$\begin{aligned}{}^L\boldsymbol{\omega}_\mathbf{m}(t) &= {}^L\boldsymbol{\omega}_\mathbf{true}(t) + {}^L\mathbf{b_g}(t) + \mathbf{n_g}(t) \\ {}^L\mathbf{a_m}(t) &= \mathbf{C_q}^\top({}^G\mathbf{a_{true}}(t) - {}^G\mathbf{g}) + {}^L\mathbf{b_a}(t) + \mathbf{n_a}(t).\end{aligned} \tag{4}$$

For the gyroscope and accelerometer measurements, we assume zero-mean white Gaussian noise $(\mathbf{n_g}(t), \mathbf{n_a}(t))$ and zero-mean first-order random walk $({}^L\mathbf{b_g}(t), {}^L\mathbf{b_a}(t))$. The estimated state $\mathbf{x}(t)$ is propagated using the following kinematic equations

$$\begin{aligned}{}^G_L\dot{\hat{\mathbf{q}}}(t) &= \frac{1}{2}\mathbf{\Omega}(\hat{\boldsymbol{\omega}}(t)){}^G_L\hat{\mathbf{q}} \qquad (5)\\ {}^L\dot{\hat{\mathbf{b}}}_\mathbf{g}(t) &= \mathbf{0}_{3x1}\end{aligned}$$

$$\begin{aligned} {}^G\dot{\hat{\mathbf{v}}}(t) &= \mathbf{C}_{\hat{\mathbf{q}}} \cdot {}^L\hat{\mathbf{a}}(t) + {}^G\mathbf{g} - 2\lfloor {}^G\boldsymbol{\omega}_{\boldsymbol{e}} \times \rfloor {}^G\hat{\mathbf{v}}(t) \\ &\quad - \lfloor {}^G\boldsymbol{\omega}_{\boldsymbol{e}} \times \rfloor^{2G}\hat{\mathbf{p}}(t) \\ {}^L\dot{\hat{\mathbf{b}}}_{\mathbf{a}}(t) &= \mathbf{0}_{3x1} \\ {}^G\dot{\hat{\mathbf{p}}}(t) &= {}^G\hat{\mathbf{v}}(t) \end{aligned}$$

where $\hat{\boldsymbol{\omega}}(t) = {}^L\boldsymbol{\omega}_{\mathbf{m}}(t) - {}^L\hat{\mathbf{b}_{\mathbf{g}}}(t) - C_{\hat{q}}^\top \cdot {}^G\boldsymbol{\omega}_{\boldsymbol{e}}$ and $\hat{\mathbf{a}}(t) = {}^L\mathbf{a}_{\mathbf{m}}(t) - {}^L\hat{\mathbf{b}_{\mathbf{a}}}(t)$. The quaternion derivative uses the matrix operation $\boldsymbol{\Omega}$

$$\boldsymbol{\Omega}(\boldsymbol{\omega}) \quad = \quad \begin{bmatrix} -\lfloor \boldsymbol{\omega} \times \rfloor & \boldsymbol{\omega} \\ -\boldsymbol{\omega}^\top & 0 \end{bmatrix} \tag{6}$$

where $\lfloor \boldsymbol{\omega} \times \rfloor$ is the skew-symmetric matrix of $\boldsymbol{\omega}$. This propagation is carried out using the Runge-Kutta integration method RK4, and it is important to note that this provides significant improvement over lower order methods. Feeding (5) directly into an RK4 solver as coupled linear/rotational integration improves accuracy vital for long distance dead-reckoning, particularly in the linear velocity estimate.

The measurement prediction step of the Kalman filter propagates the error state and the respective state covariances. The error states are a linearized version of our physical state. These states are represented in a 15-dimensional error vector defined as

$$\begin{aligned} &\tilde{\mathbf{x}}(t) = \\ &\begin{bmatrix} \tilde{\boldsymbol{\delta\theta}}^\top(t) & {}^L\tilde{\mathbf{b}}_{\mathbf{g}}^\top(t) & {}^G\tilde{\mathbf{v}}^\top(t) & {}^L\tilde{\mathbf{b}}_{\mathbf{a}}^\top(t) & {}^G\tilde{\mathbf{p}}^\top(t) \end{bmatrix}^\top . \end{aligned} \tag{7}$$

Note that the drop in dimensionality occurs in the orientation representation. Assuming orientation errors are small, the over-constrained four dimensional quaternion is approximated by a three-dimensional vector, defined by

$$\boldsymbol{\delta}\mathbf{q} \simeq \begin{bmatrix} \frac{1}{2}\tilde{\boldsymbol{\delta\theta}} \\ 1 \end{bmatrix} . \tag{8}$$

The continuous-time linearized dynamics of the error state is written as

$$\dot{\tilde{\mathbf{x}}} = \mathbf{F}_c(\mathbf{x})\tilde{\mathbf{x}} + \mathbf{G}_c\mathbf{n}, \tag{9}$$

where $\mathbf{F}_c$ is the continuous-time error state transition matrix and $\mathbf{G}_c$ is the continuous-time noise propagation matrix:

$$\mathbf{F}_c = \begin{bmatrix} -\lfloor \hat{\boldsymbol{\omega}} \times \rfloor & -\mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ -\mathbf{C}_{\mathbf{q}} \lfloor \hat{\mathbf{a}} \times \rfloor & \mathbf{0}_3 & -2\lfloor {}^G\boldsymbol{\omega}_{\boldsymbol{e}} \times \rfloor & -\mathbf{C}_{\mathbf{q}} & -\lfloor {}^G\boldsymbol{\omega}_{\boldsymbol{e}} \times \rfloor^2 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 \end{bmatrix} \tag{10}$$

$$\mathbf{G}_c = \begin{bmatrix} -\mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & -\mathbf{C}_{\mathbf{q}} & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \end{bmatrix} . \tag{11}$$

The continuous-time matrix (10) is converted to a discrete-time matrix Φ_k using Taylor series expansion on a matrix:

$$\boldsymbol{\Phi}_k = e^{\int \mathbf{F}_c(t)dt} = \mathbf{I}_{15} + \mathbf{F}_c dt + \frac{1}{2!}\mathbf{F}_c^2 dt^2 + \frac{1}{3!}\mathbf{F}_c^3 dt^3 + ... \tag{12}$$

where $\boldsymbol{\Phi}_0 = \mathbf{I}_{15}$. This is applied to the covariance update equation

$$\mathbf{P}_{k+1|k} = \boldsymbol{\Phi}_k \mathbf{P}_{k|k} \boldsymbol{\Phi}_k^\top + \mathbf{Q}_d \tag{13}$$

where the discrete-time propagation noise matrix $\mathbf{Q}_d$ is updated by

$$\mathbf{Q}_d = \boldsymbol{\Phi}_k \mathbf{G}_c \mathbf{Q}_c \mathbf{G}_c^\top \boldsymbol{\Phi}_k^\top \cdot dt \tag{14}$$

and Q_c represents the process noise model matrix, defined as

$$\mathbf{Q}_c = \begin{bmatrix} \sigma_{g_n}^2 \cdot \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \sigma_{g_b}^2 \cdot \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \sigma_{a_n}^2 \cdot \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \sigma_{a_b}^2 \cdot \mathbf{I}_3 \end{bmatrix} . \tag{15}$$

The σ terms are found from the sensor specs of the IMU, white noise terms being σ_{g_n} and σ_{a_n} and bias stability (random walk) being $\sigma_{g_b}$and σ_{a_b}, each pair for gyroscope and accelerometer respectively.

B. Measurement Update

The measurement update step of the error-state Kalman filter is used to update the uncertainty of the state given a global correction. These global corrections are supplied to the filter from a three dimensional position correction from the LIDAR localization described in section III. The measurement directly (and only) affects the position estimate, reflected by the Jacobian $\mathbf{H} = \begin{bmatrix} \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix}$. The rest of the update follows the standard Kalman filter equations

$$\begin{aligned} \mathbf{S} &= \mathbf{H}\mathbf{P}_{k+1|k}\mathbf{H}^\top + \mathbf{R} \\ \mathbf{K} &= \mathbf{P}_{k+1|k}\mathbf{H}^\top\mathbf{S}^{-1} \\ \tilde{\mathbf{x}}_k &= \mathbf{K}\mathbf{r} \\ \mathbf{P}_{k+1|k+1} &= (\mathbf{I}_{15} - \mathbf{K}\mathbf{H})\mathbf{P}_{k+1|k}(\mathbf{I}_{15} - \mathbf{K}\mathbf{H})^\top + \mathbf{K}\mathbf{R}\mathbf{K}^\top \end{aligned} \tag{16}$$

where $\mathbf{R}$ is the noise covariance matrix of the correction and the correction residual is $\mathbf{r} = \mathbf{p}_{correction} - \tilde{\mathbf{p}}$. Although orientation is not computed or corrected from LIDAR localization, the measurement update indirectly estimates the orientation error from the position error over time. The newly computed covariance $\mathbf{P}_{k+1|k+1}$ is then used for the prediction step as described previously in section IV-A.

The last step is to update the actual state estimate. The linear terms of the state vector $\mathbf{x}$ are updated by $\mathbf{x}_{k+1} = \mathbf{x}_k + \tilde{\mathbf{x}}_k$. The orientation is updated using Quaternion multiplication

$${}^G_L\mathbf{q}_{k+1} = {}^G_L\,\mathbf{q}_k \otimes \delta\mathbf{q}_{k+1}. \tag{17}$$

C. Filter Stability

Filters are susceptible to linearization errors and these errors become more apparent for long-range missions. To mitigate these issues and maintain stability, we apply the following three techniques.

First, the LIDAR localization algorithm requires time to process and when the position correction is received by the filter, the state has already been propagated multiple times. Therefore, the filter is rolled back to the state at which the

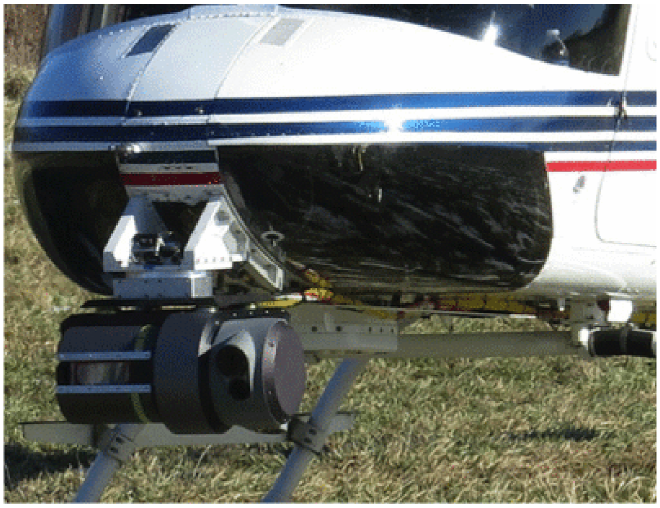

Fig. 4: The Near Earth Autonomy m4 sensor suite on a Bell 206L helicopter.

correction occurred, applied with the correction, and then the propagation is recomputed. We keep a history of states that is longer in duration than the amount of processing time required for the LIDAR localization.

Second, numerical stability is affected by the limits on the floating-point precision of the machine. Over time, small numerical errors cause the covariance matrix to diverge, causing non-optimality. To prevent this, the Joseph form of the Kalman filter measurement update is used

$$\begin{aligned}\mathbf{P}_{k+1|k+1} &= (\mathbf{I}_{15} - \mathbf{KH})\mathbf{P}_{k+1|k} \\ &= (\mathbf{I}_{15} - \mathbf{KH})\mathbf{P}_{k+1|k}(\mathbf{I}_{15} - \mathbf{KH})^\top + \mathbf{KRK}^\top.\end{aligned} \tag{18}$$

Lastly, the covariance matrix $\mathbf{P}$ must always be symmetric. While the Joseph form of the Kalman update does help prevent numerical issues, we ensure the covariance matrix maintains symmetry by regularly re-symmetrizing using

$$\mathbf{P} := \frac{1}{2}(\mathbf{P} + \mathbf{P}^\top) \tag{19}$$

at every correction step.

V. Experimental Results

A. Hardware Setup

Our datasets were collected from a Bell 206L (LongRanger) helicopter outfitted with the Near Earth Autonomy m4 sensor suite (see Fig. 4) which includes a 2D LIDAR and a strapdown fiber-optic IMU. The LIDAR scans have a 100 degree FOV providing 42,000 point measurements per second with a max range of 1.1 km. Additionally, a GPS inertial solution is collected to serve as ground truth in our evaluation. All sensors are time synchronized. The initial vehicle position and orientation in ECEF is provided, as well as the position and orientation uncertainty.

We use publicly available 10 m resolution DEMs from USGS[1] as prior elevation maps to localize against. To build a local 3D elevation model from LIDAR, we accumulate 350 sequential scans. It is important to note that this number can be adjusted based on vehicle speed to ensure a sufficiently large scan region. As shown in Table I, the vehicle averaged at about 50 m/s (180 km/hr).

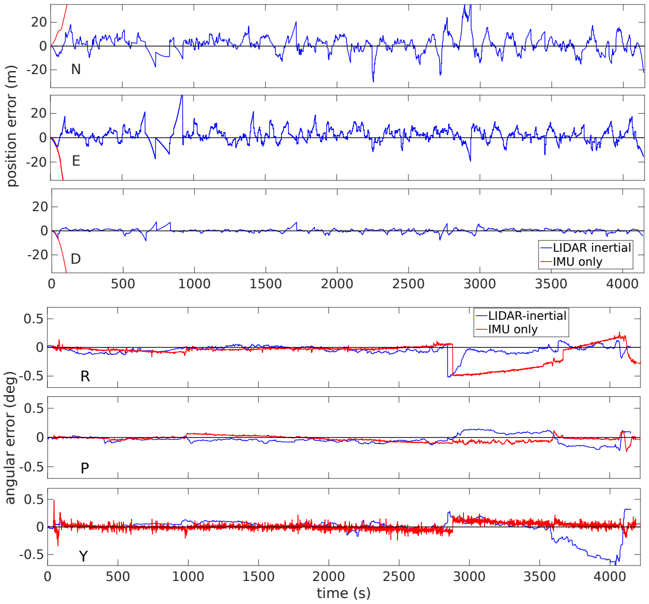

Fig. 5: Linear and angular errors of the state estimate for flight B. Our LIDAR-inertial solution (blue) is compared against an IMU-only dead reckoning attempt (red).

We tested on a quad-core 2.5GHz i7 processor, running two parallel threads for filtering and matching separately. The filter is updated at 200 Hz, taking a constant 0.1 ms for propagation, while the less frequent LIDAR measurement updates take 11 ms because of numerical stability considerations described in section IV-C. The time required for the LIDAR localization algorithm is on average 22 ms with a standard deviation of 3 ms for the presented results.

B. Evaluation

The fusion of IMU with LIDAR localization helped to reduce drift and to maintain a good state estimate for periods of poor localization performance. Fig. 6 shows the matches for the entire trajectory of flight B, with an inset showing a period that had very few corrections due to buildings, flat terrain, and a river. The rest of the dataset shows very frequent successful matches, despite the mostly dense vegetation encountered.

Table I shows the success ratio for two separate flights (recorded during the summer with dense vegetation). The LIDAR matching succeeded at least 70% of the time. The robustness of the Kalman filter is tested during longer periods of unsuccessful matches. The system kept the state estimate within 100 m of ground truth throughout the entirety of both trajectories. Fig. 5 plots the errors in individual axes against the ground truth. The blue line is the LIDAR-inertial solution estimate and the red line represents the IMU-only dead reckoning attempt over the same duration. Although

[1] http://nationalmap.gov/elevation.html

TABLE I: Statistics for two long distance flights, including LIDAR localization and state estimate performance.

Flight	Trajectory distance	Avg. flight speed	Cruising altitude range	# of match attempts	% of match successes	Longest duration without a match	Landing position error	Max. position error	RMSE
A	196 km	44.5 m/s	210-340 m	853	83.4%	62.8 s	38.6 m	90.2 m	4.22 m
B	218 km	55.6 m/s	150-250 m	851	74.1%	92.2 s	27.2 m	42.4 m	2.91 m

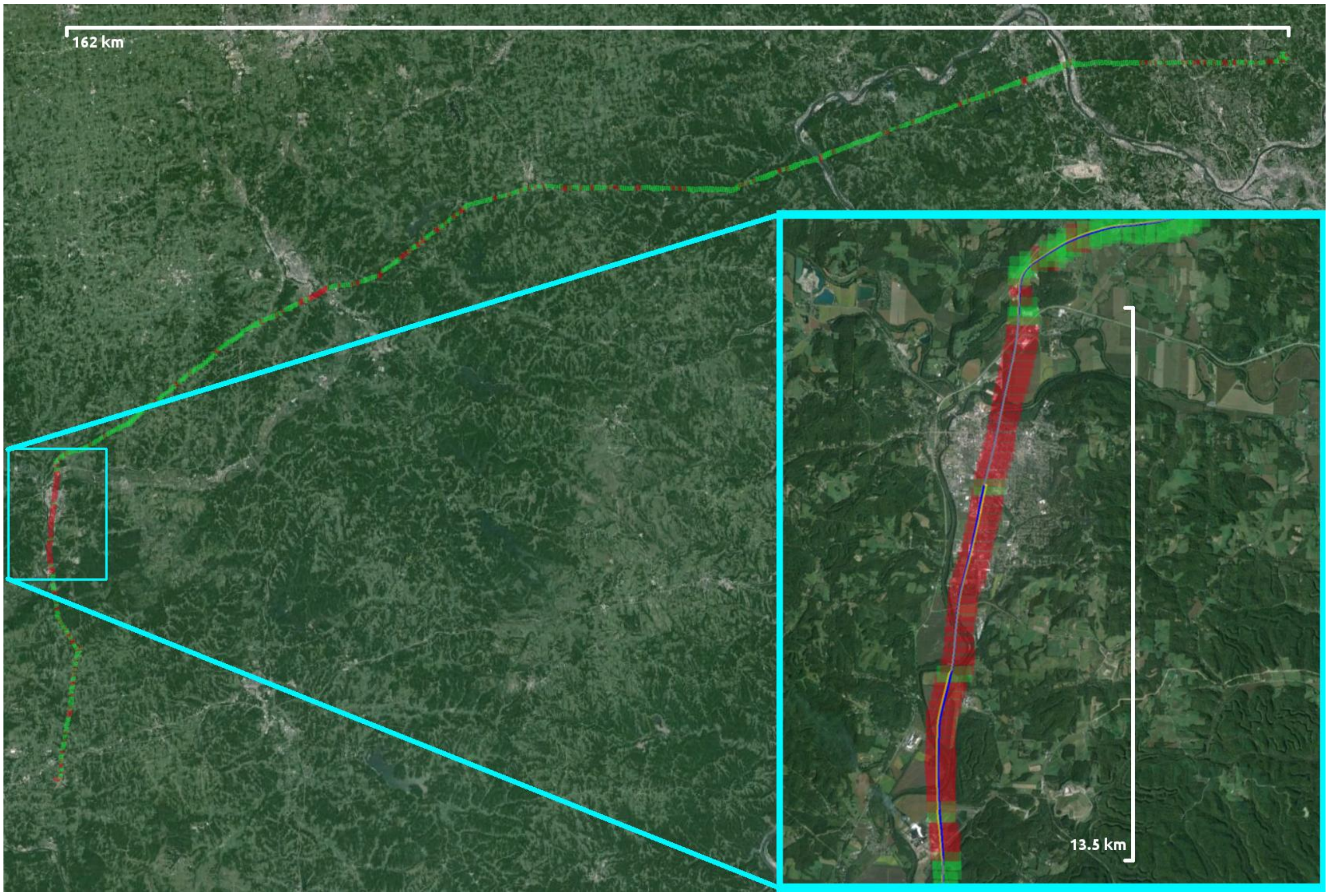

Fig. 6: LIDAR localization match successes (green rectangles) and failures (red rectangles) for the entire trajectory of flight B. The inset shows an enlarged section of the trajectory in which very few successes occurred and therefore relied on the bias-corrected inertial solution. At this scale, the ground truth trajectory (blue line) is nearly identical to the estimated trajectory (yellow line) and just barely visible in the inset.

the maximum position error between both datasets is 90.2 m, the solution stays within a 20 m error laterally and 5 m error vertically for most of the flight. Our maximum position error of 90 m is comparable to the results found in [5], however we demonstrate smaller position errors on average over the entire trajectory, as well as significantly longer distances achieved.

Fig. 7 shows Euclidean position error in relation to successful and failed matching events. As expected, during flight segments in which we fail to successfully match the LIDAR with the terrain, the position error grows over time. The growth is determined by the robustness of the IMU bias estimate and state propagation. Note that we stop acquiring successful matches near the end of the trajectory. This is due to the landing sequence, where the vehicle is too close to the ground to observe the terrain shape. Before that point, the error is less than 20 m from ground truth, while the final position error is close to 40 m.

We visualize the resulting percentage of error over distance flown in Fig. 8, comparing it to an IMU-only dead reckoning estimate. The flight lands with a distance error ratio of 0.019% over the entire 196 km trajectory. The data shows that the position error is bounded independent of the length of the flight. In contrast, the IMU-only solution drifts without bounds, showing that even for a high quality fiber-optic IMU, an IMU-only solution is infeasible and integration of additional information such as from our LIDAR localization is needed.

VI. Conclusion and Future Work

We presented a solution to long-distance aerial state estimation for LIDAR-based sensor systems. Using available a priori DEM, we are able to match a LIDAR scan region against this elevation model efficiently in real-time. By using an error-state Kalman filter, we can estimate IMU biases and provide a better position estimate. This allows us to reduce

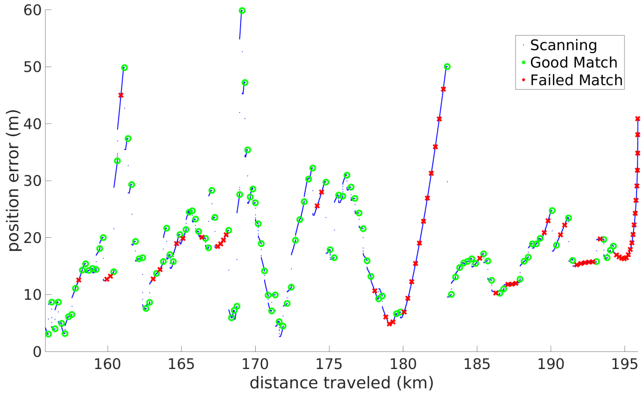

Fig. 7: Euclidean position error for the last 40 km of flight A, with successful (green) and failed (red) matching attempts marked. Each blue point represents a single 2D line LIDAR scan mapped at the vehicle's current position error, i.e. after a successful match we can see the increase or decrease of the position error by finding the start of the next blue line. The drift from the inertial estimate grows the position error over periods of failed LIDAR matches until a successful match resets the error.

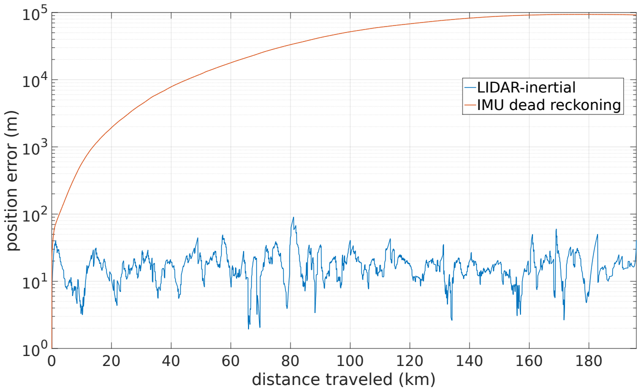

Fig. 8: Euclidean position error over the entirety of flight A. We compare our solution against an IMU-only estimate which quickly drifts away from the correct position. Our LIDAR-inertial solution keeps the position error bounded independent of distance.

the search region of the DEM as well as fly longer stretches without LIDAR position updates.

Additional sensors can be added to improve robustness or reduce requirements on IMU quality. Drift depends on both the quality of the IMU and the accuracy of the bias estimation. To improve bias estimation accuracy, visual odometry can be fused into the system. This would provide continuous bias estimate corrections for periods of unsuccessful LIDAR localization possibly due to flight over water, low terrain variability, or low-altitude flight. Therefore less drift is expected over time, allowing for longer duration missions in a wider range of terrains.

The 10 m resolution DEM provides too coarse of a correction estimate if a smooth trajectory is desired. We have tested our method on a 3 m resolution DEM over shorter trajectories and have achieved tighter position estimates. The associated 10-fold increase in memory requirements necessitate dynamic DEM management solutions to cover long trajectories.

Kalman filters are susceptible to linearization errors, but there are additional techniques beyond the presented work that can improve their performance. The discrete-time error-state transition matrix $\mathbf{\Phi}_k$ is currently approximated using the Taylor series expansion of $\mathbf{F}_c$. To increase accuracy and reduce computational cost a closed-form solution to $\mathbf{\Phi}_k$ can be derived as in Weiss et al. [11]. Another option is to use a fixed-lag smoother, allowing to revise past state estimates and therefore also improving the current state.

Acknowledgment

The authors would like to thank Jeffrey Mishler, Adam Stambler, and Marcel Bergerman at Near Earth Autonomy.

References

[1] M. U. de Haag, A. Vadlamani, J. L. Campbell, and J. Dickman, "Application of laser range scanner based terrain referenced navigation systems for aircraft guidance," *Electronic Design, Test, and Applications, Proceedings of the Third IEEE International Workshop*, 2006.

[2] J. Farrell, *Aided navigation: GPS with high rate sensors*, ser. Electronic Engineering. McGraw-Hill New York, 2008.

[3] F. Hover, R. Eustice, A. Kim, B. Englot, H. Johannsson, M. Kaess, and J. Leonard, "Advanced perception, navigation and planning for autonomous in-water ship hull inspection," *Intl. Journal of Robotics Research*, vol. 31, no. 12, pp. 1445–1464, Oct. 2012.

[4] V. Indelman, S. Williams, M. Kaess, and F. Dellaert, "Information fusion in navigation systems via factor graph based incremental smoothing," *Journal of Robotics and Autonomous Systems, RAS*, vol. 61, no. 8, pp. 721–738, Aug. 2013.

[5] A. Johnson and T. Ivanov, "Analysis and testing of a lidar-based approach to terrain relative navigation for precise lunar landing," in *AIAA Guidance, Navigation, and Control Conference*, Portland, Oregon, Aug. 2011.

[6] J. Lewis, "Fast template matching," in *Vision Interface*, Quebec City, Canada, May 1995, pp. 120–123.

[7] E. Quist, "UAV navigation and radar odometry," Ph.D. dissertation, Bringham Young University, 2015.

[8] A. R. Runnalls, P. D. Groves, and R. J. Handley, "Terrain-referenced navigation using the IGMAP data fusion algorithm," in *ION Annual Meeting*, Jun. 2005, pp. 976–987.

[9] N. Trawny, A. I. Mourikis, S. I. Roumeliotis, A. E. Johnson, and J. F. Montgomery, "Vision-aided inertial navigation for pin-point landing using observations of mapped landmarks: Research articles," *Journal of Field Robotics*, vol. 24, no. 5, pp. 357–378, May 2007.

[10] M. Warren, P. Corke, and B. Upcroft, "Long-range stereo visual odometry for extended altitude flight of unmanned aerial vehicles," *Intl. Journal of Robotics Research*, vol. 35, pp. 381–403, Apr. 2016.

[11] S. Weiss, M. W. Achtelik, S. Lynen, M. C. Achtelik, L. Kneip, M. Chli, and R. Siegwart, "Monocular vision for long-term micro aerial vehicle state estimation: A compendium," *Journal of Field Robotics*, vol. 30, no. 5, pp. 803–831, Sep. 2013.

[12] J. Zhang and S. Singh, "Visual-inertial combined odometry system for aerial vehicles," *Journal of Field Robotics*, vol. 32, no. 8, pp. 1043–1055, Dec. 2015.