Autonomous Robots
https://doi.org/10.1007/s10514-019-09899-4

Relative navigation of autonomous GPS-degraded micro air vehicles

David O. Wheeler[1] · Daniel P. Koch[2] · James S. Jackson[2] · Gary J. Ellingson[2] · Paul W. Nyholm[2] · Timothy W. McLain[2] · Randal W. Beard[1]

Received: 22 May 2018 / Accepted: 26 December 2019

Abstract
Unlike many current navigation approaches for micro air vehicles, the relative navigation (RN) framework presented in this paper ensures that the filter state remains observable in GPS-denied environments by working with respect to a local reference frame. By subtly restructuring the problem, RN ensures that the filter uncertainty remains bounded, consistent, and normally-distributed, and insulates flight-critical estimation and control processes from large global updates. This paper thoroughly outlines the RN framework and demonstrates its practicality with several long flight tests in unknown GPS-denied and GPS-degraded environments. The relative front end is shown to produce low-drift estimates and smooth, stable control while leveraging off-the-shelf algorithms. The system runs in real time with onboard processing, fuses a variety of vision sensors, and works indoors and outdoors without requiring special tuning for particular sensors or environments. RN is shown to produce globally-consistent, metric, and localized maps by incorporating loop closures and intermittent GPS measurements.

Keywords Aerial robotics · GPS-denied · Navigation · GPS-degraded · Observable

1 Introduction

Economists anticipate that autonomous micro air vehicles (MAVs) will give rise to a handful of billion-dollar markets, including infrastructure inspection, security, precision agriculture, transportation, and delivery (PricewaterhouseCoopers 2016). Using MAVs to inspect bridges, dams, chemical plants, and refineries is particularly motivating as it would take the place of dangerous, time consuming, and expensive human inspections; however, these markets are still largely speculative because autonomous MAV navigation is an active research problem, especially in confined, unknown environments where global positioning system (GPS) measurements are unavailable or degraded.

Current MAV navigation approaches rely heavily on GPS for estimation, guidance, and control; however, GPS signals can be spoofed, jammed, or blocked by structures and foliage. GPS measurements can be further degraded by multipath, atmospheric delays, or poor positioning of visible satellites. When GPS is unavailable, the MAV's global position and heading is not observable (Martinelli 2012; Weiss et al. 2012; Jones et al. 2007). As a result, when the global state is tracked directly in a filter the estimates eventually drift, leading to filter inconsistency and non-optimal sensor fusion (Bailey et al. 2006; Bar-Shalom et al. 2002). Significant reliability issues arise when working with respect to a globally-referenced state during prolonged GPS dropout and heading uncertainty (Wheeler et al. 2018).

Many current GPS-denied MAV systems utilize a Kalman filter at the core of the navigation solution. Despite the issues discussed above, the majority of these solutions estimate and control with respect to a single, inertial reference frame: either the GPS origin or the MAV's initial pose. This formulation is convenient; however, there are several underlying issues that commonly arise in GPS-degraded environments when tracking the global state directly in a filter and controlling with respect to those estimates:

- Controlling with respect to the unobservable global state precludes any guarantee on the stability of the system.
- In the absence of global measurements, estimates of the unobservable global state drift over time and the uncertainty grows without bound. If GPS is reacquired after a prolonged period of dropout and used as an update in

✉ Daniel P. Koch
daniel.p.koch@gmail.com

[1] Department of Electrical and Computer Engineering, Brigham Young University, Provo, UT 84602, USA

[2] Department of Mechanical Engineering, Brigham Young University, Provo, UT 84602, USA

Published online: 09 January 2020

the filter, the global state may jump considerably. This jump, if not accounted for, may in turn produce extreme control inputs (Weiss et al. 2012; Shen et al. 2014; Chambers et al. 2014; Tomic et al. 2012; Scherer et al. 2015). A large global uncertainty also reduces the filter's ability to properly reject degraded GPS measurements, causing the state estimate to degrade.

– During prolonged GPS dropout, the unobservable global filter states become inconsistent (Bailey et al. 2006; Wheeler et al. 2018), resulting in a poor understanding of the uncertainty of the vehicle's global pose. Inconsistency reduces estimator optimality (Bar-Shalom et al. 2002), can cause the estimator to gate valid GPS measurements if GPS is eventually reacquired, and can negatively impact applications such as geofencing that require a good understanding of the global uncertainty.

While various methods have been introduced in the literature to help mitigate or work around these issues, ultimately the root cause is unobservability.

This paper uses the recently proposed relative navigation (RN) framework (Leishman et al. 2014) as an alternative, locally-observable approach for GPS-denied MAV navigation. By using a view matcher, such as camera-based visual odometry (Scaramuzza and Fraundorfer 2011; Fraundorfer and Scaramuzza 2012) or laser-based scan matching (Bachrach et al. 2011; Gutmann and Schlegel 1996), the relative navigation filter estimates the MAV's state with respect to its local environment, while global state reconstruction is relegated to a lightweight pose-graph back end that concatenates these local estimates. The relative state estimator ensures that the state being tracked by the filter is observable and that the uncertainty remains bounded, consistent, and normally-distributed (Wheeler et al. 2018), better satisfying the assumptions underlying a Kalman filter approach. By removing the global-state estimation from the front end, RN also ensures that large or delayed global-state updates, which come from incorporating loop-closure constraints or eventual global measurements, do not impact the flight-critical control and estimation feedback. While the global state is still inherently unobservable in the absence of global measurements, consistency is maintained by estimating the global state outside the filter with the back-end pose graph, where the non-Gaussian uncertainties can be better represented. In addition, robust optimization methods can identify and reject gross GPS outliers and false-positive loop closures.

One of the primary contributions, and more novel aspects, of the RN approach is the architecture that confines all functions critical to safe flight in a relative front end which operates on observable states, while relegating all global estimation and mission planning to a back end which can better handle the non-Gaussian uncertainties and which is not subject to the strict timing requirements that apply to the flight-critical components. This decoupling also prevents jumps in global state estimates from destabilizing the vehicle. The advantages of this approach in terms of observability and consistency have been previously investigated by Wheeler et al. (2018). This paper focuses primarily on the practical advantages and implementation aspects of the architecture, and on hardware demonstrations of its effectiveness. The contributions of the paper are twofold:

Fig. 1 MAV smoothly navigating through a GPS-degraded environment

First, the details necessary to implement the complete RN framework are presented, including the subtleties of the interactions between the relative front end and global back end. Specifically, we describe the relative estimator reset operation necessary to maintain observability, and present the relative guidance and control strategy necessary to ensure smooth, stable flight. We discuss how to reconstruct the global state with consistent banana-shaped uncertainty distributions, and describe how to incorporate GPS and loop-closure information to improve the global state estimate. We explain how the high-level path planner facilitates autonomous missions and show how to leverage off-the-shelf algorithms for visual odometry, place recognition, and robust pose-graph optimization.

The second contribution consists of several prolonged hardware flight tests demonstrating the effectiveness of RN for autonomous GPS-degraded MAV navigation in varied, unknown environments, such as that shown in Fig. 1. We demonstrate that the relative front end successfully fuses multiple vision sensors, works indoors and outdoors, and results in low drift with no state jumps. We further demonstrate the onboard generation of a globally-consistent, metric, and localized map by identifying and incorporating loop-closure constraints and intermittent GPS measurements. Using this map, we demonstrate the fully-autonomous completion of mission objectives, including performing a position-hold about a global position waypoint while in a GPS-denied environment.

Section 2 reviews current state-of-the-art methods for GPS-degraded MAV navigation and Sect. 3 overviews the relative navigation framework. Sections 4 and 5 describe the components of the relative front end and global back end of the RN architecture respectively. In addition to outlining each component's role, the specific algorithms used for the hardware implementation are also presented. Section 6 describes the experimental flight tests, including the hardware and test procedures, while Sect. 7 describes the flight test results. Finally, Sect. 8 summarizes the contributions of the paper.

2 Related work

Because of the many applications of MAVs in GPS-denied and GPS-degraded environments, significant research has been performed in improving the capability and robustness of state estimation in these situations. Much of this work builds upon the simultaneous localization and mapping (SLAM) literature, but is adapted for MAVs. The full SLAM problem involves concurrently estimating the position of surrounding landmarks while reconstructing the vehicle's complete trajectory; however, due to the strict size, weight, power, and timing requirements associated with autonomous MAV operation, the SLAM problem is often simplified when applied to MAVs, only solving for the current pose of the vehicle and surrounding landmarks.

Early work demonstrated indoor MAV flight and provides approaches for many MAV navigation problems such as mapping, path planning, and control of GPS-denied multirotor platforms for short indoor trajectories. Grisetti et al. (2010) and Grzonka et al. (2012) present a *graph-based SLAM* approach to leverage laser scan-matching constraints, while Bachrach et al. (2011) and Bachrach et al. (2010) fuse scan matching data with inertial measurements in an extended Kalman filter (EKF), demonstrating a *vision-aided navigation* solution. Blösch et al. (2010) uses an EKF to track the global pose of individual landmarks, demonstrating a successful *EKF-SLAM* approach.

Some more recent work in this area has focused on improving the consistency of pose estimation without global measurements, extending the length of autonomous trajectories, and diversifying the environments in which MAVs can operate. Chowdhary et al. (2013) demonstrated a successful GPS-denied monocular vision-aided inertial navigation system (INS) including autonomous landing and takeoff. Scaramuzza et al. (2014) were the first to demonstrate prolonged (350 m) autonomous MAV flight in a GPS-denied environment. Their work used a single monocular camera for onboard stabilization and control. Shen et al. (2014) introduced a method for simultaneously fusing multiple relative view-matchers to increase robustness in difficult environments and demonstrated autonomous flight on a prolonged (440 m) indoor-outdoor flight. They used a stochastic-cloning filtering approach (Roumeliotis and Burdick 2002), which is designed to better propagate uncertainty but allows the global state covariance to grow unbounded in the absence of global measurement updates. Scherer et al. (2012) presented a graph-based state estimation system that fuses visual odometry, inertial measurements, and intermittent GPS information. The relative navigation approach shares many ideas with this approach, but removes the pose-graph optimization from the flight-critical path by additionally incorporating a front-end estimator.

Each of these previously mentioned methods ultimately track the unobserved global state. As shown by Wheeler et al. (2018), methods that directly estimate the global state are susceptible to inconsistency and state jumps during prolonged GPS dropout. The value of a relative parameterization is well-documented in the full-SLAM literature (Bailey et al. 2006; Moore et al. 2009; Sibley et al. 2009; Chong and Kleeman 1999; Kim et al. 2010), but has not been fully applied to MAV navigation. Moore et al. (2009) noted the limitations of using either a body-fixed or a globally-fixed reference frame for ground vehicles, and proposed using a local frame in which the vehicle moves smoothly. Bailey et al. (2006) showed that estimating the vehicle and landmark location with respect to a global coordinate frame results in inconsistency as heading uncertainty increases, and asserted that submapping was the only method at the time of publication for implementing consistent large-scale EKF-SLAM . Relative submapping methods (Chong and Kleeman 1999; Kim et al. 2010) estimate the state of the vehicle and landmarks with respect to a local coordinate frame. These submaps are subsequently fused and form a more consistent global estimate. Sibley et al. (2009) proposed a completely relative bundle-adjustment formulation for incrementally solving the full SLAM problem in constant time. In essence, relative navigation demonstrates how to apply these relative ideas discussed in the full-SLAM literature to computationally constrained and dynamically unstable MAV platforms using an EKF to ensure smooth flight in GPS-degraded environments.

3 Relative navigation overview

The intuition behind relative navigation is straightforward. An alert driver can safely navigate indefinitely, even if completely lost or disoriented. This is because humans instinctively perceive the world and make decisions with respect to the current local environment, as opposed to working with respect to an arbitrary global reference point. When a driver is lost, ideally an accompanying passenger looks for landmarks, references a map or GPS unit, plans the optimal global route, and then provides low-frequency, high-level

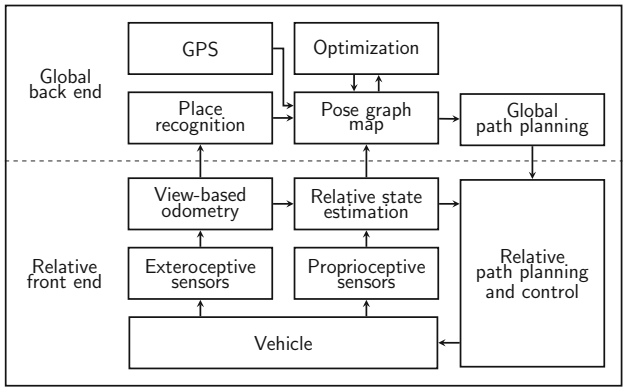


Fig. 2 Relative navigation architecture. Using relative motion measurements, such as from visual odometry or scan matching algorithms, the vehicle estimates its local state. These estimates are used for flight-critical path planning and control. As a separate process, the global back end incorporates any available global information. Its only influence on the front end is through locally-defined guidance objectives. Diagram adapted from Leishman et al. (2014)

instructions to the driver in the local frame—for example, "turn around when possible" or "make the next right turn." In this way, time- and safety-critical estimation and control decisions are decoupled from potentially erroneous global information.

Figure 2 presents the relative navigation architecture introduced by Leishman et al. (2014), where the decoupled responsibilities of the relative front end and global back end are analogous to a driver and passenger. Using relative motion measurements, available from a keyframe-based view-based odometry source such as visual odometry or scan matching, the relative front-end filter estimates its pose with respect to its local environment. This observable, relative state estimate is used for flight-critical path planning and control. The filter resets its relative states to zero each time the view-based odometry declares a new keyframe, which happens when enough motion has occured that sufficient keyframe features are no longer in view. Before resetting, the filter passes its current relative pose estimate and covariance to the back end. As a separate process, the global back end concatenates these relative states as edges in a pose-graph map, and additionally incorporates any available global information such as place recognition constraints or GPS measurements. The only way the global back end influences the front end is through locally-defined guidance objectives.

The relative navigation architecture is readily applied to existing systems, as it does not make any assumptions about the vehicle platform or sensor suite. A wide variety of algorithms can be used to implement each component, and due to the modular nature of RN, it is straightforward to interchange the algorithms as needed. The RN framework also allows multiple view-matchers to be used simultaneously for increased robustness in difficult environments. In the next two sections we describe the details of the relative front end and the global back end.

4 Relative front end

By working with respect to the local environment, the relative front end ensures that the flight-critical estimation, guidance, and control always operates with respect to an observable state, allowing smooth, stable flight even when global information is degraded or undergoing large corrections. When relative navigation was first presented by Leishman et al. (2014), the discussion emphasized a particular choice for a visual odometry algorithm, estimator, path planner, and controller. This section generalizes that discussion by outlining the fundamental nature of each front-end component, highlighting how existing algorithms would need to be adapted to fit into the relative navigation architecture. Specifically, we describe how to incorporate current state-of-the-art view-based odometry algorithms, describe the relative estimator reset operation necessary to maintain observability, and present the relative guidance and control strategy necessary to ensure smooth, stable flight.

4.1 View-based odometry

When GPS is not available, MAVs commonly use odometry measurements computed from exteroceptive sensors such as laser scanners or cameras. A variety of odometry algorithms exist including laser scan matching (Bachrach et al. 2011; Gutmann and Schlegel 1996) and visual odometry (Scaramuzza and Fraundorfer 2011; Fraundorfer and Scaramuzza 2012). In the context of the RN framework, these view-based odometry algorithms output the incremental or relative transform between the two frames (images or scans), without performing any type of global localization. While some odometry methods compare consecutive frames, others compare the current frame to a recent *keyframe*. When a keyframe is used, a series of odometry measurements are computed with respect to this keyframe. Generally the keyframe is updated only when there is insufficient overlap to provide a reliable odometry measurement. As a result, keyframe-based odometry reduces temporal drift in the computed odometry as compared to frame-to-frame matching (Shen et al. 2014; Leutenegger et al. 2015).

In contrast to the current work, which uses only the incremental transform computed by the view-matching algorithm as an input to the filter (similar to other approaches such as that of Shen et al. (2014)), some work (such as that by Scaramuzza et al. (2014) and Weiss and Siegwart (2011)) uses the term *visual odometry* to refer to a black-box algorithm that provides pseudo-global position updates to the filter. In this case the odometry algorithm is performing additional

steps to concatenate and optimize the incremental transforms to produce an estimate of the exteroceptive sensor's global pose. Such implementations exceed the scope of what is required as an input to the RN framework, since we perform this global concatenation in the back-end pose graph. However, the excellent work that has gone into these implementations can often be leveraged in the RN framework by isolating the portion of the algorithm that computes the relative updates between frames, while removing the global portions. This is the approach we used with the DEMO RGB-D odometry algorithm (Zhang et al. 2014) and with the CSM scan-matching algorithm (Censi 2008) for our flight test results.

As shown in Fig. 2, the view-matcher is only loosely coupled to the estimator. As such, it is straightforward to accept relative measurements from any source or sensor, such as RGB-D and stereo visual odometry or a laser scan matcher. Monocular cameras introduce the additional difficulty of translational scale ambiguity in the visual odometry solution. While some work has been done on addressing this difficulty by combining the relative navigation approach with tightly-coupled visual-inertial estimation techniques (Jackson et al. 2019), the architecture and results presented in this paper are restricted to odometry measurements with known scale factors computed from sensors with depth information.

The framework can additionally handle multiple relative sensors, which can increase the robustness of the system in difficult environmental conditions. For example, Koch et al. (2016) demonstrated using RN to simultaneously incorporate relative measurements from a laser scanner and RGB-D camera. While the scan matcher broke down in long hallways and the visual odometry broke down in a dark room, the redundant sensing allowed the vehicle to successfully navigate these environments.

As mentioned above, for the results in Sect. 7 we used relative adaptations of DEMO (Zhang et al. 2014) for visual odometry and CSM (Censi 2008) for scan matching.

4.2 Relative state estimation

Most MAV navigation approaches continue to estimate the global state, even when GPS-dropout makes the global state unobservable. Given an inertial measurement unit, altimeter, and even visual odometry measurements, the global position and heading of a MAV in the horizontal plane cannot be observed (Martinelli 2012; Weiss et al. 2012). With time, the associated state estimates drift and become inconsistent.

One fundamental advantage of RN is that the front-end state always remains observable when accurate relative odometry measurements are available. RN maintains observability by defining the state with respect to a local *node frame*. The node frame is defined as the gravity-aligned coordinate frame that is positioned on the ground exactly under the MAV when the current keyframe was taken. Because each node frame is gravity-aligned and positioned on the ground, the MAV's altitude, roll, and pitch (z, ϕ, θ) with respect to the node frame are estimated no differently than if defined with respect to a global origin. By referencing the current node frame, however, the horizontal position and yaw states (x, y, ψ) now correspond to the relative position and heading of the MAV with respect to the most recent odometry keyframe. In this way, relative measurements provided by a view-matcher directly measure the MAV's relative position and heading, causing the state to be observable by construction provided there are a sufficient number of features distributed throughout the field of view to produce accurate measurements. With regular, direct updates, the uncertainty of the vehicle's relative state remains consistent, bounded, and approximately Gaussian (Wheeler et al. 2018).

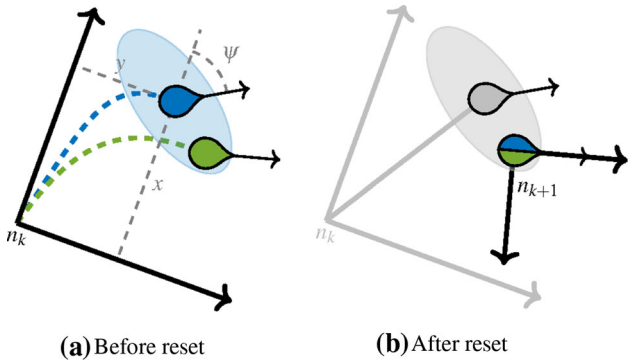


Fig. 3 2D illustration of node frame reset operation. **a** The relative estimator tracks the MAV's position and heading (x, y, ψ) with respect to the current node frame n_k. The estimated state (blue) will not perfectly align with the true MAV state (green), but the estimator's covariance (blue oval) should correctly represent the underlying uncertainty. **b** When a new keyframe is declared, the new node frame n_{k+1} is defined at the true, yet globally unknown, MAV pose. The estimated pose (gray) and covariance (gray oval) are saved as an edge constraint in the back-end pose graph and the MAV's (x, y, ψ) states and their corresponding covariance terms are reset to zero (Color figure online)

A variety of estimation techniques are used for MAV navigation and could be adapted to become a relative estimator. The fundamental concept is that the estimator's state and covariance should be reset whenever a new keyframe is declared. Figure 3 illustrates the process of transitioning from one keyframe to the next. The relative estimator tracks the MAV's position and heading (x, y, ψ) relative to the current node frame n_k. Naturally the estimated state will not perfectly align with the true MAV state, but the estimator's covariance should correctly represent the underlying uncertainty. When a new keyframe is declared, the new node frame n_{k+1} is defined at the true, yet globally unknown, MAV position. The current pose and covariance are saved as an edge constraint in the back-end pose graph and then the MAV's position and heading states and their corresponding covariance terms are

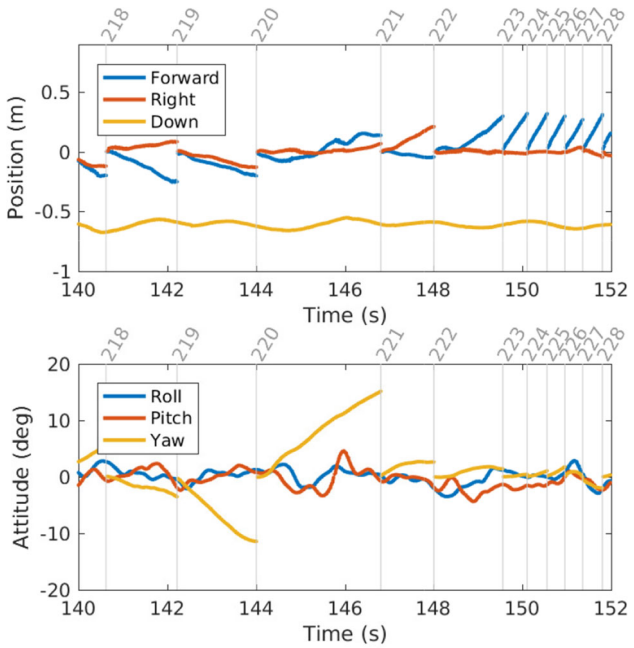


Fig. 4 Typical mid-flight state estimates. The vertical gray lines indicate a new node frame, and the labels indicate the associated node identifier. With each new node frame, the forward, right, and yaw states are reset to zero, while the down, roll, and pitch states are unaffected. The vehicle was yawing from 142 s to 146 s and moving forward at a constant velocity from 150 s to 152 s. While the state estimates are discontinuous, the relative navigation approach facilitates smooth, stable navigation in GPS-degraded environments

reset to zero. In this way, the global uncertainty is removed from the front-end filter and delegated to the global back end.

Figure 4 shows example state estimates, where the horizontal position and heading states are reset at each new node frame. While the discontinuities in the state estimates may appear concerning from a control perspective, they occur at known times and thus are reliably handled by the relative path planner and controller to produce smooth, stable control. It is important to note that while the front-end filter tracks the full six degrees-of-freedom pose, it is sufficient to only optimize the relative states (x, y, ψ) in the back end.

For the flight results described in Sect. 7, we used an indirect formulation of the multiplicative extended Kalman filter as presented by Koch et al. (2017). A unit quaternion is used to represent the MAV's attitude while attitude error is propagated using a minimal three-state representation. When a new keyframe is declared, care is taken to only reset the unobserved horizontal position and heading, leaving roll, pitch, altitude and their associated uncertainties unchanged. Refer to the work by Koch et al. (2017) for additional estimator details including the state, dynamics, sensor models, and specific details about the reset step.

Some similarities and differences exist between the RN relative estimator and the popular robocentric estimation approach. As described by Koch et al. (2017), a relative estimator can be defined in either an inertial or robocentric representation. A robocentric approach tracks the position of landmarks with respect to the current body's pose. While robocentric dynamics are less intuitive, this formulation ensures improved observability and consistency properties for the landmark states, similar to RN. Most robocentric approaches, however, continue to track the global state of the MAV with respect to the current body. After prolonged flight without global information, the global uncertainty is not well represented by a Gaussian distribution in typical Cartesian coordinates, leading to estimator inconsistency (Wheeler et al. 2018). The relative navigation framework provides a method to use either inertial or body-fixed dynamics, produces smooth, observable state estimates for control, and represents the global state consistently.

4.3 Relative path planning and control

Within the relative navigation framework, all front-end guidance and control is computed with respect to the current node frame. Many current MAV controllers drive the estimated global state to a desired global state. These same controllers can be directly applied to drive the estimated relative state to a desired relative state. Any control approach can be used as long as care is taken to ensure that the estimator and controller are working with respect to the same reference frame.

Each time the relative estimator resets to a new node frame, the path planner and controller must also update to ensure that they are operating with respect this new frame. Depending on the chosen control strategy, this update operation may range from updating an entire potential field to requiring no action as in the case of a body-fixed velocity controller.

Let $\mathbf{x}^c_{a/b}$ represent the state a with respect to frame b, expressed in frame c. Using this notation, Fig. 5 illustrates

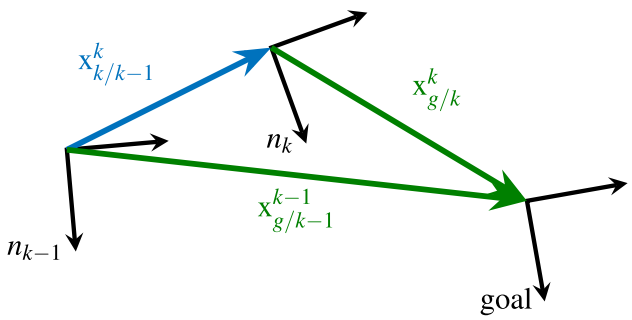


Fig. 5 Updating a relative goal when a new node frame is declared. The goal with respect to the previous keyframe, $\mathbf{x}^{k-1}_{g/k-1}$, is expressed with respect to the new keyframe, $\mathbf{x}^k_{g/k}$, using the edge constraint $\mathbf{x}^k_{k/k-1}$ provided by the relative estimator

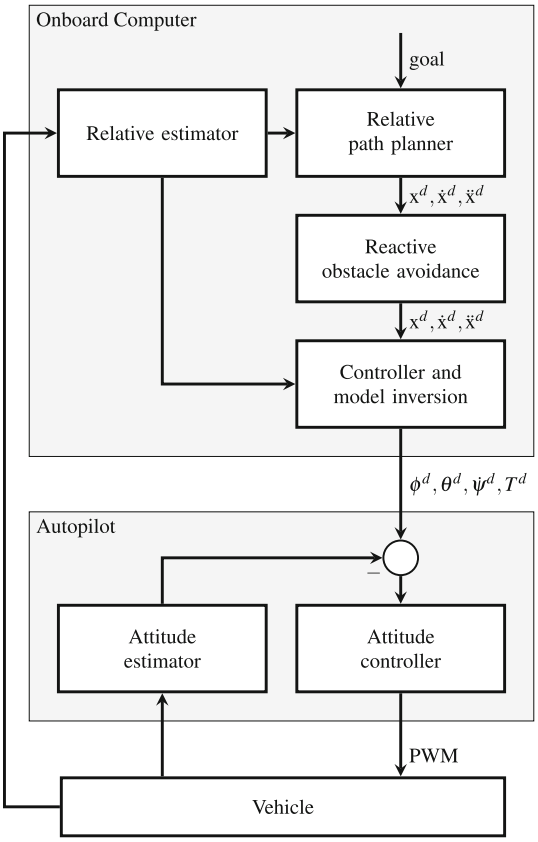


Fig. 6 Control architecture. The autopilot performs high-rate feedback control about roll, pitch, yaw rate, and thrust commands provided from the onboard computer. Diagram adapted from Wheeler et al. (2016)

the process of updating a position goal that is expressed with respect to the previous node frame n_{k-1} to the current node frame n_k. In short, the relative path planner uses the estimated edge constraint between subsequent node frames provided by the relative estimator, $\mathbf{x}_{k/k-1}^{k-1}$, to express the previous goal $\mathbf{x}_{g/k-1}^{k-1}$ in the new node frame, $\mathbf{x}_{g/k}^{k}$. Because each node frame is gravity-aligned and positioned on the ground, any roll, pitch, altitude, or body-fixed velocity components of the goal remain unchanged.

As a practical note, we recommend that the relative controller incorporate logic to monitor if the relative estimator's node frame identifier matches the node frame identifier of the current goal. If the node frame identifiers are not in sync or no goal is supplied by the path planner, the MAV is directed to hover in place. While this step is an important safety precaution, the controller did not enter this state during our flight testing. With a careful implementation, the control performance does not degrade due to the relative state reset.

Figure 6 presents the control architecture used to avoid collisions and produce the smooth, flight-critical control needed to safely operate the MAV in unknown, dynamic environments with unpredictable external disturbances. The onboard computer uses its current relative state estimate and a path planning algorithm to calculate a trajectory to the current relative goal. We use the reactive obstacle avoidance plugin framework (Jackson et al. 2016b) to use the latest sensor information to modify the current trajectory when needed to avoid a pending collision. Control loops are then closed around this modified trajectory to produce desired accelerations. At this point, the non-linear model of the MAV dynamics is inverted (Michael et al. 2010; Raffo et al. 2010), providing a desired roll, pitch, yaw rate, and thrust command. These attitude setpoints are passed to the autopilot where high-rate attitude feedback control is performed.

For the results in Sect. 7, the path planner uses position feedback to supply high-rate velocity goals. These velocity goals are then modified using the cushioned extended-periphery obstacle avoidance algorithm (Jackson et al. 2016b). An LQR feedback controller is closed around the modified velocity setpoints to produce desired accelerations, which are then passed through the model inversion to produce the roll, pitch, yaw rate, and thrust command that is sent to the autopilot.

5 Global back end

While the relative front end ensures flight-worthiness, if a MAV is tasked with performing a global mission then a global state estimate is required. This section describes how the global state and its uncertainty are reconstructed. While the overall concept of the RN back end was presented previously by Leishman et al. (2014), the implementation details presented in this section are unique contributions of this paper.

5.1 Pose-graph map

Before resetting the state and establishing a new node frame, the front end saves the estimated relative pose and associated uncertainty. Because each node frame is defined to be located at the true (yet globally unknown) position of the MAV, the uncertainty is reset with each node frame. This ensures that the saved pose estimates from one node frame to the next are mutually independent. This facilitates structuring the back end as a *pose graph*.

A pose graph is a conventional graph where each vertex or node corresponds to the global pose of a vehicle at a certain instant in time, and graph edges represent the relative change in position and attitude from one node to another. Odometry measurements, such as the relative pose estimates from the relative estimator, provide edge constraints between sequential nodes. If a *place recognition* algorithm detects that the vehicle has returned to a previous pose, an edge constraint between non-consecutive poses, known as a *loop closure*, is introduced in the graph. The vehicle's global pose can be

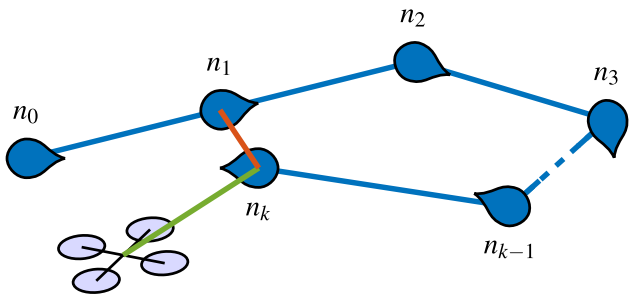


Fig. 7 To reconstruct the MAV pose with respect to the origin, each estimated edge (blue line) is compounded, followed by the MAV state with respect to the current node (green line). A loop-closure constraint (red line) in general will not perfectly agree with the odometry constraints, resulting in an over-constrained system (Color figure online)

reconstructed by first traversing the graph from the origin to the current node, compounding each estimated edge in the path, and then incorporating the relative state. When loop closures are added, the graph is over-constrained and multiple paths, and therefore multiple pose estimates, are possible. This is illustrated in Fig. 7. A weighted-least-squares optimization can be performed to reconcile these discrepancies, removing accumulated drift. Other, more involved frameworks leverage the *factor graph* data structure which uses Bayesian methods to infer the pose of the MAV over time by representing edge constraints as *factors*. Factor graph methods have the added benefit of being able to solve for the global uncertainty of each pose and can incorporate other measurements such as range-only or IMU preintegration factors (Forster et al. 2015; Carlone et al. 2014). Both factor-graph and pose-graph formulations are able to solve for the optimal set of poses given odometry and loop-closure edge constraints with associated uncertainties.

Formulating the back-end optimization problem as a pose graph results in the following beneficial properties:

- A variety of well-developed pose-graph optimization frameworks exist to find a consistent global representation of the trajectory after accounting for all constraints (Kummerle et al. 2011; Kaess et al. 2008; Dellaert and Kaess 2006; Kaess et al. 2012).
- Robust pose-graph optimization techniques can identify and remove the effect of erroneous constraints such as false-positive loop closures or degraded GPS (Sunderhauf and Protzel 2013; Agarwal et al. 2013; Latif et al. 2013).
- A pose-graph representation provides a straightforward method to consistently represent a MAV's global state uncertainty. When global measurements are unavailable, representing error using the vector space formed by the Lie algebra $\mathfrak{se}(3)$ produces banana-shaped, Gaussian uncertainty distributions that better parameterize the underlying distribution (Wheeler et al. 2018; Barfoot and Furgale 2014; Long et al. 2012). The ability of the pose-graph representation to reconstruct this distribution was extensively explored in the context of the RN framework by Wheeler et al. (2018).
- A pose graph provides a lightweight representation of a trajectory, ensuring scalability and practicality on resource-constrained platforms or networks. Long trajectories with a large number of loop closures can benefit from node removal techniques which further reduce the complexity of the optimization problem (Carlevaris-Bianco et al. 2014).

Pose graphs are commonly used for MAV back ends; however, many approaches that track the global state in the front end do not provide a clear method to construct independent edge constraints and covariances, an issue addressed explicitly by relative navigation.

5.2 Place recognition

An important aspect of pose-graph back ends is the ability to remove accumulated drift if the MAV detects that it has returned to a previously visited location. Place recognition algorithms efficiently compare the current keyframe image or scan to each previous keyframe image or scan. When a strong correspondence is detected, the relative transformation is computed and an edge constraint between non-consecutive nodes, known as a loop closure, is included in the pose graph.

Place recognition is a challenging problem, but a variety of approaches have been successfully demonstrated (Lowry et al. 2016). To scale well, the method must be fast and efficient. Additionally, the algorithm should correctly detect loop closures when there are partial occlusions, varied viewpoints or lighting conditions, or minor scene changes. It should also correctly avoid perceptual aliasing, which is falsely correlating nearly identical, yet non-unique, scenes such as two similar brick walls.

To ensure scalability, many approaches use a bag-of-words approach (Sivic and Zisserman 2003; Nister and Stewenius 2006). Salient features are identified in a representative set of images and are clustered to form a set of common, yet visually distinct, image features. This precomputed set of features, known as the *vocabulary*, is then used to describe each vehicle image. Using a common vocabulary allows for a sparse representation and facilitates rapid comparison. Commonly, hierarchical trees are also used for quicker comparisons. Some methods use the estimated global uncertainty to limit the set of past images that are compared.

While any place recognition algorithm could be used, we use fast, appearance-based mapping (FAB-MAP), a linear-complexity algorithm that uses Bayesian probabilities to infer the likelihood of a match while explicitly rejecting perceptual aliasing in the environment (Cummins and Newman

Fig. 8 Example loop closure detected using FAB-MAP between keyframe 80 and keyframe 416 during flight test 2

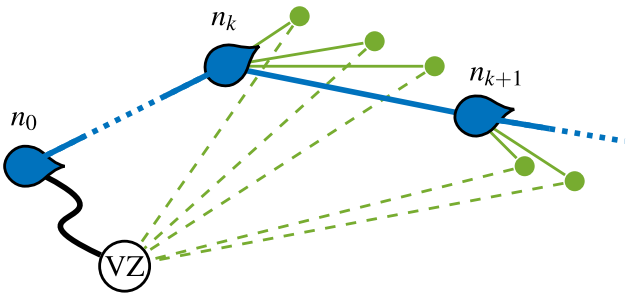


Fig. 9 Back-end GPS integration method. For each GPS measurement, one node and two edge constraints are added. The new node (green circle) is related to the virtual-zero node using the measurement and uncertainty reported by the GPS receiver (dashed green line), and is related to the current node frame using the current relative state estimate (solid green line). A virtual constraint with maximum uncertainty is added between the first node and the virtual zero node to ensure connectedness (black line) (Color figure online)

2011; Glover et al. 2012). This appearance-based matching technique provides only an image pair, so the RGB-D visual odometry algorithm described by Leishman et al. (2014) is used to calculate the full six degrees-of-freedom transform between the two images. This algorithm uses RANSAC (Fischler and Bolles 1981) to find the transform between the RGB-D image pair, and the number of outliers in the RANSAC model can be used to filter false loop closures. With this method, no false loop closures have been detected in the entirety of our flight-testing experience, and it has been shown to be computationally tractable on a MAV. An example loop closure is shown in Fig. 8.

5.3 GPS integration

While loop closures and odometry can be used in a pose graph formulation to produce a metric map of previous states, globally-referenced measurements, such as GPS, can be used to localize the map in the global frame and further improve global-state estimation. Measurements to landmarks with known global positions can also be used to localize the map globally. For example, while the results presented in Sect. 7 do not use any *a priori* information, it is trivial to seed the place recognition algorithm with a set of geo-located images.

Many MAV navigation methods estimate the global state in the front end and can directly fuse global measurements. This works well when global information is regularly available and accurate, but is shown to lead to inconsistency when the estimates drift during prolonged GPS dropout (Bailey et al. 2006; Wheeler et al. 2018). Furthermore, directly applying a global measurement to remove drift induces a large state update, often causing the control effort to jump which can destabilize the system (Weiss et al. 2012; Shen et al. 2014; Chambers et al. 2014; Tomic et al. 2012; Scherer et al. 2015). Several methods have been proposed to address this, such as simultaneously tracking a GPS-corrected and odometry-only global trajectory (Shen et al. 2014) or using a series of measurement gates (Chambers et al. 2014).

Alternatively, global measurements can be handled exclusively in the back-end pose graph using a *virtual-zero* node. Described by Rehder et al. (2012) and Scherer et al. (2012), the virtual-zero node represents the GPS origin. To ensure the pose graph is fully connected, an arbitrary edge constraint with infinite uncertainty, known as the *virtual constraint*, is applied between the virtual-zero node and the node representing the MAV's origin. For each GPS measurement received, one node and two edge constraints are added to the pose graph, as shown in Fig. 9. A node is added to represent the current vehicle pose. This node is related to the virtual-zero node using the measurement and uncertainty reported from the GPS, and is related to the current node frame using the current relative state estimate. Upon optimizing the pose graph after the first GPS measurement, the virtual constraint will correctly estimate the global position of the MAV's starting point. Incorporating subsequent GPS measurements will refine this position estimate and provide a heading estimate for the MAV's starting point, causing the entire pose graph map to be globally localized. Similar concepts have been used to incorporate multiple agents with unknown initial starting points (Kim et al. 2010).

In practice, pose graph optimizers are less likely to diverge when all constraints are of a similar order of magnitude. GPS constraints are challenging because the GPS origin is generally far away. To address this issue, we save the initial GPS measurement and subtract it from each GPS measurement before adding the edge constraint. As a result, the virtual zero constraint represents the position of the first node with respect to the first GPS measurement, as opposed to representing the position of the first node with respect to the GPS origin. If it is necessary to express the pose graph in a global coordinate frame, such as for visualizing the graph on an ortho-rectified image, the initial GPS measurement is simply added to each pose.

There are several significant advantages of using pose graphs for incorporating GPS measurements. First, due to the decoupled nature of the relative navigation framework, global state jumps cannot degrade flight-critical control. This also means that processing or networking delays can be tol-

erated. Second, robust optimization techniques can be used to detect erroneous GPS measurements. Once detected, any negative effect is completely removed from the system. Such a claim is not possible using conventional, front-end filtering methods. Finally, as few as two global measurements can be leveraged to localize the pose graph map, a research problem originally motivated by Rehder et al. (2012).

5.4 Optimization

Pose graph optimization is formulated as a weighted least-squares problem. The objective of the optimization is to find the set of global poses $\mathbf{x}$ for each node such that the set of relative edge constraints ξ are best satisfied collectively. Edge constraints are partitioned into three sets: odometry constraints $\mathscr{O}$, loop-closure constraints $\mathscr{L}$, and GPS constraints $\mathscr{G}$. Each edge constraint ξ_{ij} has an associated information matrix Ω_{ij} to represent the confidence of the constraint connecting nodes i and j. A particular estimate of global node poses $\hat{\mathbf{x}}$ can be used to determine the currently estimated relative relationship between nodes:

$$\hat{\xi}_{ij} = \mathbf{h}_{ij}(\hat{\mathbf{x}}).$$

Using this notation, the optimization is formulated as

$$\hat{\mathbf{x}}^* = \text{argmin}_{\hat{\mathbf{x}}} \sum_{\xi_{ij} \in \{\mathscr{O}, \mathscr{L}, \mathscr{G}\}} (\mathbf{h}_{ij}(\hat{\mathbf{x}}) - \xi_{ij})^\top \Omega_{ij} (\mathbf{h}_{ij}(\hat{\mathbf{x}}) - \xi_{ij}).$$

Before loop-closure and GPS constraints are introduced into the system, the optimization problem is not overconstrained and a zero-cost, odometry-only trajectory is available. When additional constraints are added, the optimization works to modify the trajectory, particularly adjusting the portions of the trajectory with the greatest uncertainty.

Pose-graph optimization is a well-researched problem. The optimization is commonly solved using iterative Gauss–Newton techniques. First, the global position of each node is estimated, often using the odometry-only trajectory. Then, for each iteration, the cost function is linearized about the current state estimate and the optimal state update for the linearized system is computed and applied. There are several known issues with this method that are addressed in the literature:

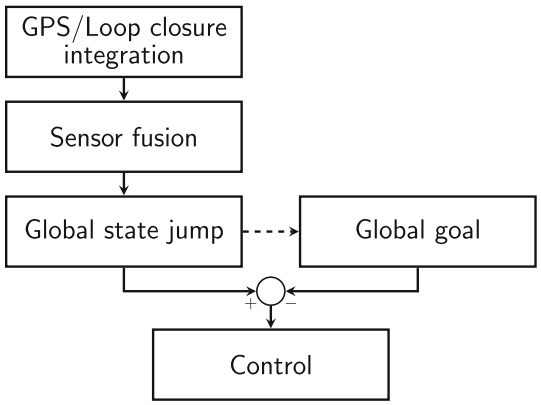
- A naive implementation requires large matrix inversions and therefore does not scale well. However, several popular pose-graph optimization frameworks have been presented that leverage sparse matrix properties and show improved scalability (Kummerle et al. 2011; Kaess et al. 2008).
- Gauss–Newton approaches can converge to a local minimum or even diverge, particularly when the initial state estimate is poor, which is common for drifting MAVs in GPS-denied environments. Several approaches have been presented to address initialization issues, including by Olson et al. (2006).
- Least-squares optimization is highly sensitive to outliers. While outliers are unlikely for odometry constraints, false-positive loop-closure constraints or degraded GPS measurements can significantly impact the optimization. Switching constraints (Sunderhauf and Protzel 2013), dynamic covariance scaling (DCS) (Agarwal et al. 2013), max-mixture models (Olson and Agarwal 2013) and the RRR algorithm (Latif et al. 2013) are all proven methods for detecting outliers and mitigating their effect on the optimization.

While these and similar methods help prevent the back end from diverging, they do not guarantee convergence, nor do they necessarily provide smooth or timely global-state estimates. This further highlights the importance of decoupling flight-critical processes from global information. For the flight-test results in Sect. 7 we used g2o (Kummerle et al. 2011) with dynamic covariance scaling (Agarwal et al. 2013).

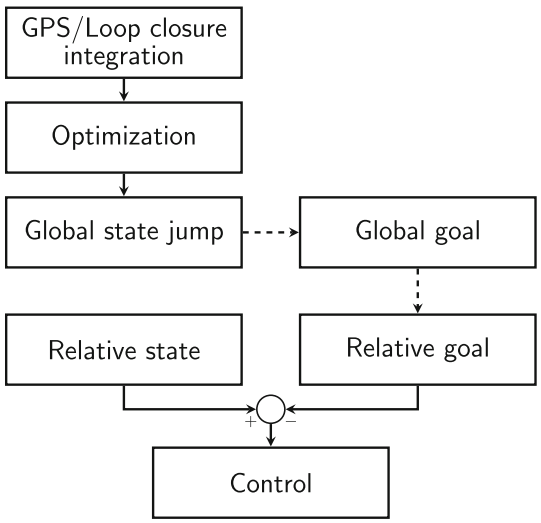
5.5 Global path planning

The role of the global path planner is first to determine the optimal MAV trajectory by assessing relevant global information, and second to transform the plan to be with respect to the current node frame for use in the relative front end. A variety of path planning algorithms could be used depending on the mission objective, including autonomous exploration, mapping, target tracking, waypoint following, cooperative control, or landing. After a plan is determined, the global path planner passes relative goals to the relative path planner. When a new keyframe is declared, these goals are updated to be expressed with respect to the latest node frame. These relative goals are the only way the global back end influences the MAV, which helps isolate the front end from destabilizing or erroneous global information. This idea is illustrated in Fig. 10.

A simple global path planner was implemented for the flight test results in Sect. 7. Since the MAV begins without any global information, a user initially takes the place of the global path planner by supplying a series of position or velocity setpoints. After the MAV travels for some distance and creates a global map, the user specifies a desired global waypoint on the map. At this point, Dijkstra's algorithm is used to search through the back-end pose graph to find the shortest known path to the desired waypoint. The global path planner then supplies velocity setpoints to the relative front end to direct the MAV along the path to the global waypoint. This method is sufficient for autonomous MAV navigation in unknown environments and demonstrates the role of the



(a) Typical global state estimation approach



(b) Relative navigation

Fig. 10 Illustrations of how incorporating global information influences vehicle control. The columns respectively represent estimation and planning, and the dashed arrows indicate optional relationships. **a** Introducing global information into a conventional approach causes a global state jump which directly influences control (Weiss et al. 2012; Shen et al. 2014; Chambers et al. 2014; Tomic et al. 2012; Scherer et al. 2015). **b** With the relative navigation approach, a global state jump never affects the relative state estimate. Vehicle control is only influenced as the global path planner provides an updated relative goal to the relative path planner

global path planner, but more sophisticated planners could be implemented for other mission scenarios.

6 Experimental setup

The experimental platform, shown in Fig. 11, is a hexacopter with a diameter of 0.69 m through the prop centers and a mass of 4.8 kg. The vehicle carries a 3DR Pixhawk autopilot, onboard computer, IMU, RGB-D camera, planar laser scanner, GPS receiver, and ultrasonic altimeter. The details

Fig. 11 The vehicle used for the flight tests

Table 1 Hardware details

Component	Description
Platform	Hexacopter, 4.8 kg, 0.69 m diameter
Autopilot	3DR Pixhawk
RGB-D Camera	ASUS Xtion Pro Live
Laser Scanner	Hokuyo UTM-30LX
IMU	MicroStrain 3DM-GX3-15
Altimeter	I2CXL-MaxSonar-EZ MB1242
GPS	U-blox LEA-6T
Processor	Intel Core i7-2710QE (2.1 GHz × 4)
Memory	8GB DDR3

of the hardware configuration are summarized in Table 1. It is important to note that the purpose of this research is to demonstrate a successful framework for GPS-degraded MAV navigation and not to meet a particular specification or optimally address a specific application. We selected common sensors, processors, and algorithms without much consideration for optimizing the MAV's size, weight, speed, or endurance.

The data flow and networking between the various system components are illustrated in Fig. 12. The relative navigation framework was implemented entirely on the onboard computer in C++ using the Robot Operating System (ROS) (Quigley et al. 2009) middleware. Attitude control was performed by a 3DR Pixhawk autopilot running a customized version of the PX4 firmware[1]. During fully autonomous sections of flight, a ground station laptop was used to send waypoint commands to the onboard computer over Wi-Fi via the ROS messaging system. During semi-autonomous sections of flight, velocity commands were sent to the onboard computer by a human operator using a wireless Microsoft Xbox controller. At all times, a human safety pilot had a direct RC link to the Pixhawk autopilot to override attitude commands from the onboard computer if necessary. Safety pilot

[1] The PX4 firmware is customized to accept inputs from the onboard computer while also allowing an RC safety pilot to override these commands if necessary. We have subsequently transitioned to using the ROSflight autopilot (Jackson et al. 2016a); see https://rosflight.org.

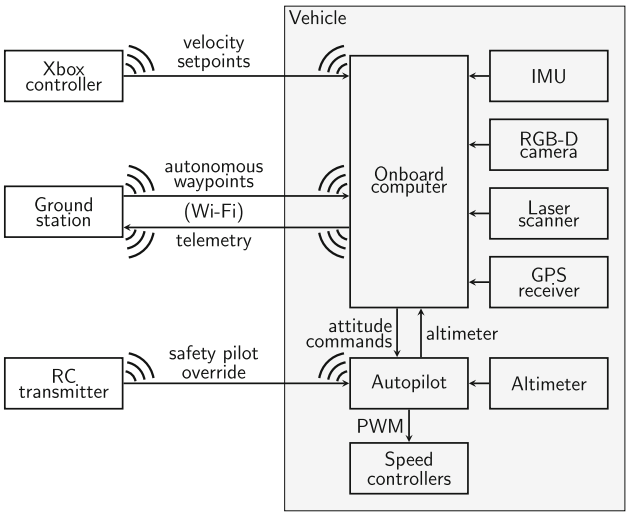


Fig. 12 The data flow and networking between the various system components

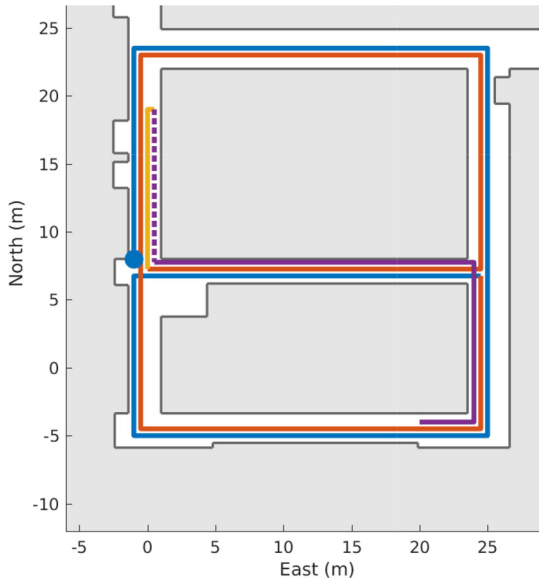


Fig. 13 Flight test 2. The vehicle started at the blue circle moving clockwise, following the blue path, red path, yellow path, and then purple path. The vehicle flew in the middle of the hallway and was facing the direction of motion except for the path indicated by purple dots when the vehicle traveled backwards (Color figure online)

intervention was not required during the flight tests described in this paper.

The following three flight tests demonstrate autonomous MAV navigation in a variety of challenging unknown environments using the relative navigation framework. All perception, estimation, control, and mapping was performed onboard the vehicle and in real time. Estimation and control were performed at the rate of the IMU measurements, which was 100 Hz. Visual odometry was performed at 15 Hz using the RGB-D camera, and laser-scan matching was performed at 40 Hz. No adjustments or tuning were required to prepare the vehicle for the different scenarios other than choosing between the RGB-D camera and laser scanner, illustrating that the framework does not make environment-specific assumptions. The flight tests are described in the following sections, and are summarized in Table 2. A discussion of the results demonstrated by these flight tests is given in Sect. 7.

6.1 Flight test 1: outdoor GPS-denied

In the first flight test, the vehicle flew a trajectory around the perimeter of a large building, marked in black in Fig. 17. The flight lasted 9 min, and the total distance traveled was 320 m. For this flight test the system obtained visual odometry from the RGB-D camera. A human operator provided velocity setpoints to the vehicle through the Xbox controller. Because the MAV flew within a few meters of the building throughout the flight, reliable GPS measurements were not available. Because the vehicle did not revisit any portion of the flight path, loop-closure constraints were also unavailable.

6.2 Flight test 2: indoor GPS-denied

This flight test was conducted indoors through a series of hallways. The flight path of the vehicle is overlaid on the floor plan of the building in Fig. 13. The flight lasted 12 min, and the total distance traveled was 390 m. Visual odometry was obtained using the RGB-D camera. The odometry was of high quality throughout most of the flight, but its accuracy degraded in the southeast corner when the camera was pointed at a blank wall. A human operator provided velocity setpoints to the vehicle using the Xbox controller to guide the vehicle through the hallways. A total of 139 loop closures were detected using the RGB-D camera. This flight test was originally attempted by Leishman et al. (2014); however,

Table 2 Summary of flight tests

Flight Test	Environment	Distance	Duration	Sensor	GPS	Loop closures	Nodes	Figures
1 (Sect. 6.1)	Outdoor (dusk)	320 m	9 min	RGB-D	Denied	0	491	17
2 (Sect. 6.2)	Indoor	390 m	12 min	RGB-D	Denied	139	659	13,14,16,18
3 (Sect. 6.3)	Indoor/outdoor (night)	240 m	9 min	laser	Intermittent	30	891	1, 15, 19

Fig. 14 Large MAV smoothly navigating through a tight, nondescript hallway

the trajectory flown was significantly shorter, no loops were closed, and the back-end place recognition, map optimization, and global path planner had not yet been implemented.

Figure 14 shows the vehicle flying down one of the hallways. The hallways were relatively nondescript, with few visually interesting features. Despite this, the odometry and place recognition algorithms performed well. Another challenge presented by the hallways was their narrow width; the hallways ranged between only 1.8 m and 2.5 m wide, as compared to the 1.1 m total diameter of the vehicle. The narrow confines produced significant aerodynamic ground and wall effect. To highlight the significance of this effect, a highly experienced safety pilot attempted to fly the trajectory in attitude stabilized mode via RC control, and struggled to maintain stability in the wider hallways to the extent that flying in the narrower hallways was unfeasible.

6.3 Flight test 3: indoor/outdoor intermittent GPS

The third flight test consisted of two loops through both indoor and outdoor environments through and near a building. This flight test incorporated loop closures, intermittent degraded GPS, and autonomous path planning and flight into a single experiment. The flight lasted 9 min and traveled a distance of 240 m. The path that the vehicle flew is overlaid on a satellite image of the building in Fig. 15. The vehicle started inside the southeast wing of the building, flew through the courtyard into the northeast wing of the building, down the alleyway to the east of the building and back into the southeast wing, then repeated the same path. In all, there were four transitions from indoor to outdoor, and four transitions from outdoor to indoor. These transitions are commonly troublesome for GPS-degraded navigation approaches because odometry algorithms can sometimes degrade and GPS accuracy can vary significantly through the transition.

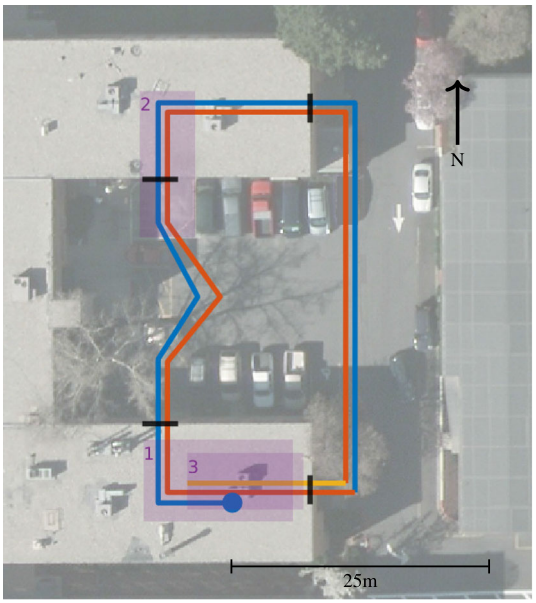


Fig. 15 Flight test 3. The vehicle started at the blue circle, moving clockwise, following the blue path, red path, and then yellow path. The vehicle was facing the direction of motion. Purple indicates regions of autonomous waypoint following and black indicates the doorways (Color figure online)

Odometry was obtained from the laser scanner, while loop closures were obtained using the RGB-D camera. The flight test was conducted at night, so loop closures were obtained only in the well-lit indoor portions. In all, 30 loop closures were detected. Due to the close proximity to the buildings, GPS updates were very limited. GPS measurements were gated until the GPS receiver's self-reported accuracy estimate dropped below a reasonable threshold. As a result, all GPS measurements were gated until the second time the vehicle flew down the alley between buildings. Even then, only ten GPS updates were received, and these updates were biased to the north by about two meters.

During the first loop, the vehicle was guided by velocity setpoints provided by a human operator using the Xbox controller. After the first loop-closure constraints were detected and the map was optimized to remove drift, fully autonomous waypoint following was demonstrated. A human operator clicked on a previously visited point on the map, and the vehicle retraced its previous path to arrive at the desired waypoint. Three of these fully autonomous segments were carried out, marked in purple in Fig. 15, including one during an outdoor to indoor transition.

In addition to the results presented in this paper, this same indoor/outdoor flight was also performed a second time during the day using the RGB-D camera instead of the laser scanner. The alternate odometry source produced comparable front-end estimation and control, introduced 45 loop-closure constraints, successfully incorporated 36 GPS measurements, and performed four autonomous waypoint

missions. This helps to highlight the modularity and extensibility of the relative navigation framework. We chose to present the laser scanner results because they demonstrate the use of a different odometry source than that used in the other two flight tests.

7 Results

This section discusses the results from the flight tests described in Sect. 6 as they relate to various aspects of the relative navigation architecture. In general, these results demonstrate that the proposed architecture runs onboard the vehicle in real time, and that it enables missions involving real vehicles in realistic environments. The results show that the system is able to operate in both indoor and outdoor environments, and handle transitions between them. Notably, no tweaking or tuning of the system was required between the flight tests other than choosing which sensor (the RGB-D camera or laser scanner) would be used for odometry. This demonstrates that the architecture does not make environment-specific assumptions, and that it is not tied to one particular sensor.

Section 7.1 discusses the estimation accuracy and consistency from using the relative navigation approach. Section 7.2 discusses the performance of the pose-graph optimization, and Sect. 7.3 discusses the capabilities for autonomous flight demonstrated by the tests.

7.1 Estimation accuracy and consistency

Figure 16 shows the pose-graph map for the first 130 m of flight test 2. Up to this point no loop closures had been detected, meaning that the pose graph simply compounds the relative edges produced by the front-end estimator to reconstruct the global pose without any additional optimization. The accuracy of this global pose therefore directly reflects the accuracy of the front-end estimator. Figure 16 shows that only 1.8 m of drift were accumulated in the first 130 m of flight, yielding a drift rate of 1.4 percent per distance traveled. For the 139 loop closures in flight test 2, the maximum drift rate was 1.5 percent with an average drift rate of 0.85 percent. For the 30 loop closures in flight test 3, the maximum drift rate was 2.8 percent with an average of 1.8 percent. The overall accuracy of an approach depends on the environment, quality of sensors and calibration, and sophistication of odometry algorithms. These flight tests highlight that RN facilitates good performance with off-the-shelf algorithms and sensors.

Another advantage of the pose-graph representation is that it accurately captures the uncertainty in the global pose of the vehicle. Approaches that estimate the global pose directly in the filter represent the uncertainty as a Gaussian

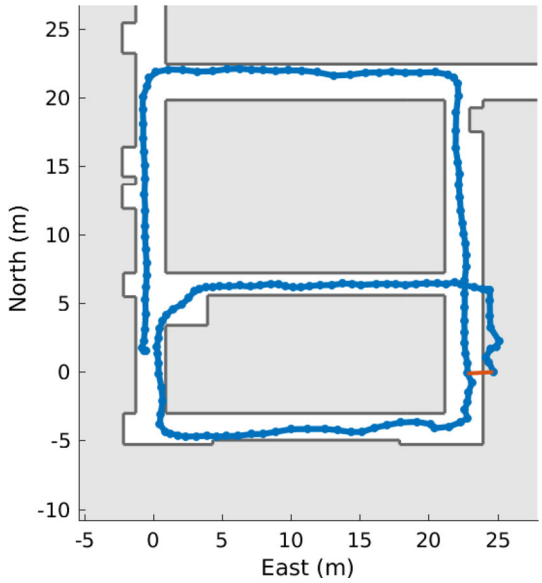


Fig. 16 Pose-graph map for the first 130 m of flight test 2. At this point the first loop closure (red) was detected and used to improve the global map without affecting local stability. Before optimization, the global pose estimate created by compounding relative edges had accumulated 1.8 m of drift (Color figure online)

normal distribution characterized by its covariance matrix, which produces an ellipsoidal confidence region. Yet, it has been shown that the true uncertainty distribution produced as a vehicle moves through the environment with uncertainty in its heading is a banana-shaped distribution (Thrun et al. 2005), which is a Gaussian distribution expressed in exponential coordinates (Long et al. 2012). A pose graph represents the global pose as a sequence of short transforms, each with an associated ellipsoidal uncertainty. It was shown by Barfoot and Furgale (2014) that this series of uncertainties can be combined to produce a total uncertainty estimate that is an excellent approximation to the true banana-shaped distribution. Therefore, the pose-graph representation contains all of the information that is necessary to produce an accurate estimate of the global pose uncertainty. Figure 17 shows the 90 percent confidence regions created from the pose graph at several points using the method presented by Wheeler et al. (2018). This method samples from the individual edge covariances in a Monte-Carlo fashion, then fits a Gaussian distribution in exponential coordinates to the resulting distribution of final pose estimates.[2] As can be seen, the resulting distributions correctly capture the banana shape of the true uncertainty distribution. In addition, at every point along the trajectory, the 90 percent confidence region captures the true location of the vehicle. This demonstrates that the uncer-

[2] Individual edge covariances can also be combined using the fourth-order analytical approximation presented by Barfoot and Furgale (2014).

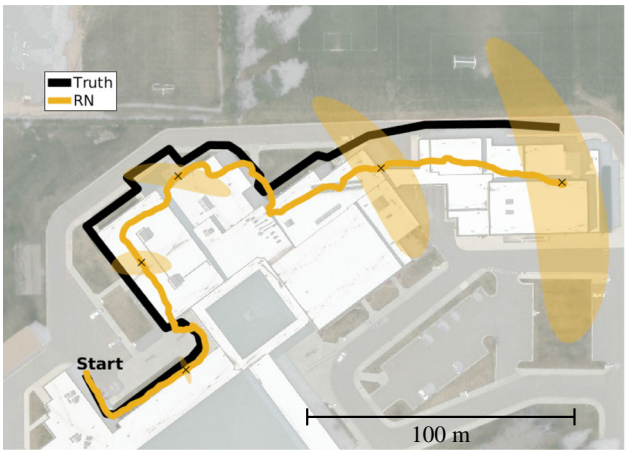


Fig. 17 Pose-graph map for flight test 1. Heading errors cause the position uncertainty to grow. The global back end compounds the small, Gaussian edge covariances to form banana-shaped uncertainty estimates that correctly represent the underlying uncertainty. The 90 percent confidence regions are shown for several instances throughout the trajectory

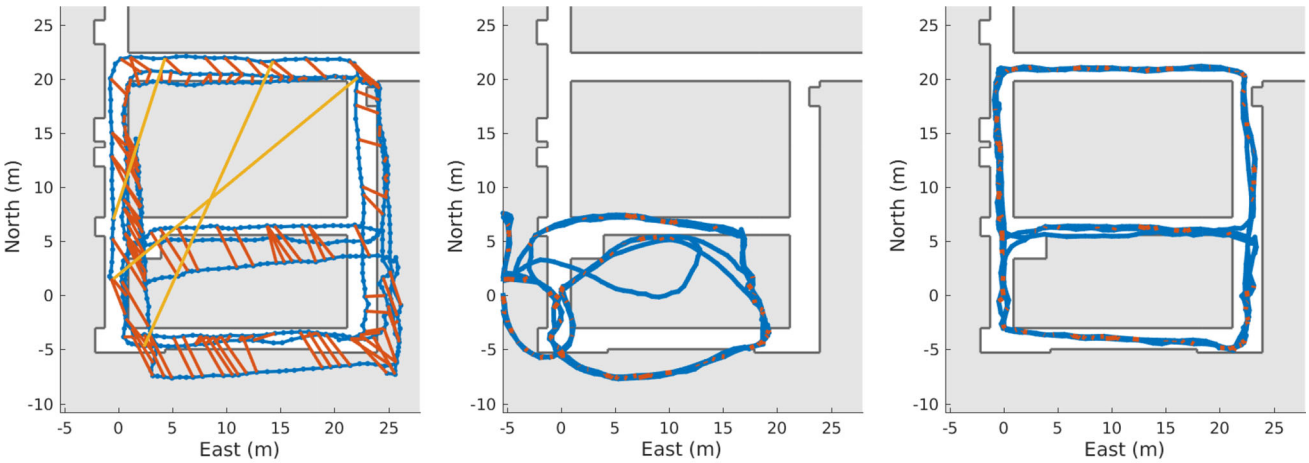
tainty estimate in the global pose reconstructed using the pose graph is consistent. More details on the consistency of the relative navigation approach, and how it compares with other state-of-the-art methods, are given by Wheeler et al. (2018).

7.2 Map optimization

Figure 18 shows the pose-graph optimization results for flight test 2. Figure 18a shows the unoptimized map produced by compounding the relative front-end pose estimates. These odometry edges are represented by the blue lines with keyframes marked as dots, and loop closures detected between keyframes are represented by red lines. Over the course of the 390 m flight, several meters of drift accumulated so that the resulting map lies outside of confines of the hallway where the vehicle actually flew. Figure 18c shows that after the map has been optimized, this drift has been removed and the estimates of the vehicle's global trajectory lie within the hallways. The complete optimization took seven iterations to converge and took less than 8 ms running onboard the vehicle during flight.

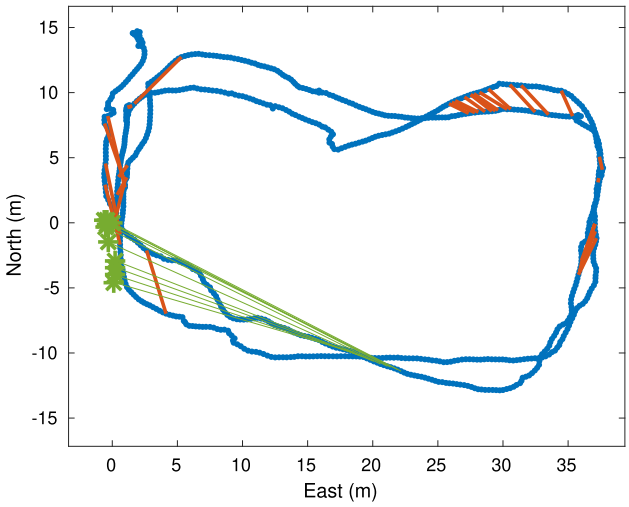
During flight test 2, the place recognition algorithm did not produce any false-positive loop-closure detections. This is particularly impressive given the fairly uniform appearance of the hallways that the vehicle flew through (see Figs. 8 and 14). To demonstrate the impact that false-positive loop-closure constraints can have, and to demonstrate the ability of the robust optimization algorithm to detect and reject these spurious constraints, three false-positive loop-closure constraints were artificially introduced to the pose graph. These are shown in yellow in Fig. 18a. Figure 18b shows the optimized pose graph obtained by a non-robust optimization algorithm that naively incorporates the false-positive constraints. The three false constraints have a drastic impact on the accuracy of the optimized map, even though there are 139 valid loop closures constraining the map. Figure 18c, on the other hand, demonstrates the effectiveness of dynamic covariance scaling in correctly detecting and rejecting the false-positive loop closures to produce a highly accurate optimized map.

The unoptimized pose-graph map for flight test 3 is shown in Fig. 19a. As with flight test 2, the relative edges from the front-end estimator are shown in blue, and the loop-closure



(a) Unoptimized trajectory **(b)** Non-robust optimization **(c)** Robust optimization

Fig. 18 Pose-graph map for flight test 2. **a** Throughout the 390 m flight 139 loop closures were detected (red) and three false-positive loop closures were artificially introduced (yellow). **b** False-positive loop closures cause a non-robust optimization to diverge. **c** Robust optimization techniques result in a consistent map. The optimization ran onboard and took 8 ms to converge (Color figure online)



(a) Pre-optimization

(b) Post-optimization

Fig. 19 Pose-graph map for flight test 3. **a** Trajectory (blue) before incorporating loop closures (red) and GPS measurements (green). For plotting purposes, the GPS is plotted relative to the first received GPS measurement. **b** After incorporating the ten available GPS measurements (green), the trajectory is globally localized. Black indicates the doorways. Note that because all of the available GPS measurements were slightly biased to the north, the final map is also biased (Color figure online)

constraints are shown in red. Again, no false-positive loop closures were detected during this flight test. In addition to loop-closure constraints, flight test 3 introduces intermittent GPS measurements. The ten valid GPS measurements are plotted as green points in Fig. 19a, and the corresponding edges in the graph are represented by green lines. As described in Sect. 5.3, the GPS constraints were defined with respect to the first GPS measurement, which is plotted at the origin. The final optimized map incorporating both loop closures and GPS measurements is shown in Fig. 19b. While truth is not available, the accuracy of the final map can be evaluated by comparing it to the satellite image of the building. The doors of the building that the vehicle flew through are marked as black lines in Fig. 19b. Due to the challenging urban canyon environment, all of the GPS measurement were biased to the north by a few meters, and so the resulting map is also biased to the north. Correcting for this bias, however, it can be seen that the optimized trajectory passes through each of the doors and matches the path that the vehicle actually flew.

One important result that this flight test demonstrates is the ability of the relative navigation architecture to perform delayed localization using few GPS points. Before the first GPS measurement is received, the map is metrically consistent with respect to the starting location of the vehicle, but is not localized globally. In other words, the vehicle knows where it is relative to its starting point, but has no knowledge of where it is in the world. This unlocalized map, however, is still sufficient for navigation purposes, and the vehicle was able to fly autonomous waypoints before it received GPS measurements. When the vehicle received the first GPS measurement, however, it was able to pin the map to a location in the world. Subsequent measurements allowed it to orient the map and refine its position estimate. For flight test 3, this localization did not occur until several minutes into the flight. In addition, the localization is accomplished using few (only ten) GPS measurements. This is significant in the context of other GPS-degraded approaches that require GPS for a prolonged (the first 80 seconds of flight) initial alignment phase (Scaramuzza et al. 2014) or have GPS for a majority of their flight (Shen et al. 2014).

7.3 Autonomous flight

A basic requirement for autonomous flight is robust and stable control of the vehicle. While difficult to quantify, the robustness of the relative navigation architecture is demonstrated by the scope of flight tests presented in this paper. For example, flight test 2 demonstrates smooth, stable flight down narrow hallways that produce significant aerodynamic ground and wall effect. The high-rate feedback control and accurate relative state estimates facilitated missions that would be infeasible for experienced human pilots. In flight test 3, the vehicle smoothly transitions through eight doorways. Between the three flight tests presented, the platform was flown for almost a kilometer through congested environments without incident. Throughout the flight tests, the

control performance did not suffer from the resetting of the relative states.

A unique advantage of relative navigation that is demonstrated by the flight-test results is the architecture's innate ability to handle jumps in the global-state estimate. For example, the pose-graph optimization at the first loop closure in flight test 2 resulted in a global state jump of 1.8 m, and the optimization at the first loop closure in flight test 3 resulted in a jump of 2.3 m. In addition, the first GPS measurements received in flight test 3 caused a large state jump as the map was rotated counterclockwise by 90 deg and translated approximately 28 m when it was first localized globally. Despite these large state jumps, the control of the vehicle did not suffer at all because, as described in Fig. 10, control is carried out in the relative frame and insulated from global state jumps by the path planner. Conceptually, this allows the MAV's perception of the local environment to remained fixed while the global map shifts beneath it.

In addition to smooth local control, flight test 3 also demonstrated autonomous global navigation. After the first loop closures were received and the drift in the map was removed, a waypoint was provided by an operator clicking on a previously visited location on the generated pose graph. The vehicle then autonomously followed the map back to this location. Autonomous waypoint following was demonstrated three times, traveling 35 m through congested environments including during an outdoor to indoor transition. The regions where this took place are highlighted in purple on Fig. 15. The final waypoint was selected after GPS measurements were incorporated into the pose-graph map. The user, by selecting a pixel on an ortho-rectified image, was effectively establishing a desired GPS waypoint for the vehicle. Of note, this global waypoint was located indoors. The vehicle then autonomously navigated to that global waypoint and stabilized its position. This result is particularly compelling because the vehicle correctly stabilized itself about a global waypoint despite being in a GPS-denied environment.

8 Conclusion

Developing dependable, autonomous MAV solutions that are robust to GPS degradation is a challenging but highly relevant field of research. This paper demonstrates that the relative navigation framework offers a compelling alternative paradigm for approaching the problem. By decoupling flight-critical estimation, guidance, and control algorithms from unobservable global states that are prone to inconsistency and state jumps, relative navigation avoids many issues that plague other state-of-the-art approaches.

This paper presents the details necessary to implement the complete relative navigation framework, including resetting the relative estimator to ensure observability and adapting existing view-matching, path planning, and control algorithms for reliable, smooth flight. We describe how to leverage pose graphs to opportunistically incorporate loop-closure and GPS constraints, and outline how the high-level path planner facilitates autonomous missions while insulating the vehicle from the negative effects of global state jumps.

Through a series of prolonged flight tests, we demonstrated the effectiveness of the relative navigation approach for autonomous GPS-degraded MAV navigation in varied, unknown environments. We showed that the system can utilize a variety of vision sensors, works indoors and outdoors, runs in real-time with onboard processing, and does not require special tuning for particular sensors or environments. We demonstrated stable front-end performance with low drift while leveraging off-the-shelf sensors and algorithms. We further demonstrated the onboard generation of a globally-consistent, metric, and localized map by identifying and incorporating loop-closure constraints and/or intermittent GPS measurements. With this map, we demonstrated the fully autonomous completion of mission objectives, including performing a position-hold about a GPS waypoint while in a GPS-denied environment.

One of the most important aspects of the relative navigation architecture is that it does not make any assumptions about a particular platform, sensor suite, environment, or use case. Many existing systems could be readily modified to fit within the relative navigation framework, and thereby benefit from its theoretical and practical advantages.

Acknowledgements This work has been funded by the Center for Unmanned Aircraft Systems (C-UAS), a National Science Foundation Industry/University Cooperative Research Center (I/UCRC) under NSF award Numbers IIP-1161036 and CNS-1650547, along with significant contributions from C-UAS industry members. This work was also supported in part by Air Force Research Laboratory Science and Technology (AFRL S&T) sponsorship. This research was conducted with Government support under and awarded by DoD, Air Force Office of Scientific Research, National Defense Science and Engineering Graduate (NDSEG) Fellowship, 32 CFR 168a. The authors would like to thank Kevin Brink of the Air Force Research Laboratory Munitions Directorate (AFRL/RW) for his support of this project and for his valuable insights.

References

Agarwal, P., Tipaldi, G. D., Spinello, L., Stachniss, C., & Burgard, W. (2013). Robust map optimization using dynamic covariance scaling. In *IEEE international conference on robotics and automation* (pp. 62–69)

Bachrach, A., He, R., & Roy, N. (2010). Autonomous flight in unknown indoor environments. *International Journal of Micro Air Vehicles*, *1*(4), 217–228.

Bachrach, A., Prentice, S., & Roy, N. (2011). RANGE-Robust autonomous navigation in GPS-denied environments. *Journal of Field Robotics*, *28*(5), 644–666.

Bailey, T., Nieto, J., Guivant, J., Stevens, M., & Nebot, E. (2006). Consistency of the EKF-SLAM algorithm. In *IEEE/RSJ international conference on intelligent robots and systems* (pp. 3562–3568).

Bar-Shalom, Y., Kirubarajan, T., & Li, X. R. (2002). *Estimation with applications to tracking and navigation*. New York: Wiley.

Barfoot, T. D., & Furgale, P. T. (2014). Associating uncertainty with three-dimensional poses for use in estimation problems. *IEEE Transactions on Robotics*, *30*(3), 679–693.

Blösch, M., Weiss, S., Scaramuzza, D., & Siegwart, R. (2010). Vision based MAV navigation in unknown and unstructured environments. In *IEEE international conference on robotics and automation* (pp. 21–28)

Carlevaris-Bianco, N., Kaess, M., & Eustice, R. M. (2014). Generic node removal for factor-graph SLAM. *IEEE Transactions on Robotics*, *30*(6), 1371–1385.

Carlone, L., Kira, Z., Beall, C., Indelman, V., & Dellaert, F. (2014). Eliminating conditionally independent sets in factor graphs: A unifying perspective based on smart factors. In *IEEE international conference on robotics and automation* (pp. 4290–4297).

Censi, A. (2008). An ICP variant using a point-to-line metric. In *IEEE international conference on robotics and automation* (pp. 19–25).

Chambers, A., Scherer, S., Yoder, L., Jain, S., Nuske, S., & Singh, S. (2014). Robust multi-sensor fusion for micro aerial vehicle navigation in GPS-degraded/denied environments. In *American control conference* (pp. 1892–1899).

Chong, K. S., & Kleeman, L. (1999). Feature-based mapping in real, large scale environments using an ultrasonic array. *International Journal of Robotics Research*, *18*(1), 3–19.

Chowdhary, G., Johnson, E. N., Magree, D., Wu, A., & Shein, A. (2013). GPS-denied indoor and outdoor monocular vision-aided navigation and control of unmanned aircraft. *Journal of Field Robotics*, *30*(3), 415–438.

Cummins, M., & Newman, P. (2011). Appearance-only SLAM at large scale with FAB-MAP 2.0. *International Journal of Robotics Research*, *30*(9), 1100–1123.

Dellaert, F., & Kaess, M. (2006). Square root SAM: Simultaneous localization and mapping via square root information smoothing. *International Journal of Robotics Research*, *25*(12), 1181–1203.

Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, *24*(6), 381–395.

Forster, C., Carlone, L., Dellaert, F., & Scaramuzza, D. (2015). *IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation* (pp. 6–15). Robotics: Science and Systems.

Fraundorfer, F., & Scaramuzza, D. (2012). Visual odometry: Part II: Matching, robustness, optimization, and applications. *IEEE Robotics and Automation Magazine*, *19*(2), 78–90.

Glover, A., Maddern, W., Warren, M., Reid, S., Milford, M., & Wyeth, G. (2012). OpenFABMAP: An open source toolbox for appearance-based loop closure detection. In: IEEE international conference on robotics and automation (pp. 4730–4735).

Grisetti, G., Kummerle, R., Stachniss, C., & Burgard, W. (2010). A tutorial on graph-based SLAM. *Intelligent Transportation Systems Council*, *2*(4), 31–43.

Grzonka, S., Grisetti, G., & Burgard, W. (2012). A fully autonomous indoor quadrotor. *IEEE Transactions on Robotics*, *28*(1), 90–100.

Gutmann, J. S., & Schlegel, C. (1996). AMOS: Comparison of scan matching approaches for self-localization in indoor environments. In: Proceedings of the 1st Euromicro workshop on advanced mobile robots (pp. 61–67).

Jackson, J., Nielsen, J., McLain, T., & Beard, R. (2019). Improving the robustness of visual-inertial extended Kalman filtering. In: *IEEE international conference on robotics and automation*.

Jackson, J. S., Ellingson, G. S., & McLain, T. W. (2016a). ROSflight: a lightweight, inexpensive MAV research and development tool. In *International conference on unmanned aircraft systems* (pp. 758–762).

Jackson, J.S., Wheeler, D.O., & McLain, T.W. (2016b). Cushioned extended-periphery avoidance: A reactive obstacle avoidance plugin. In *International conference on unmanned aircraft systems* (pp. 399–405).

Jones, E., Vedaldi, A., & Soatto, S. (2007). Inertial structure from motion with autocalibration. In *ICCV workshop on dynamical vision*.

Kaess, M., Ranganathan, A., & Dellaert, F. (2008). iSAM: Incremental smoothing and mapping. *IEEE Transactions on Robotics*, *24*(6), 1365–1378.

Kaess, M., Johannsson, H., Roberts, R., Ila, V., Leonard, J. J., & Dellaert, F. (2012). iSAM2: Incremental smoothing and mapping using the Bayes tree. *International Journal of Robotics Research*, *31*(2), 216–235.

Kim, B., Kaess, M., Fletcher, L., Leonard, J., Bachrach, A., Roy, N., & Teller, S. (2010). Multiple relative pose graphs for robust cooperative mapping. In *IEEE International Conference on Robotics and Automation* (pp. 3185–3192)

Koch, D. P., McLain, T. W., & Brink, K. M. (2016). Multi-sensor robust relative estimation framework for GPS-denied multirotor aircraft. In *International Conference on Unmanned Aircraft Systems* (pp. 589–597).

Koch, D. P., Wheeler, D. O., Beard, R. W., McLain, T. W., & Brink, K. M. (2017). *Relative multiplicative extended Kalman filter for observable GPS-denied navigation*. http://scholarsarchive.byu.edu/facpub/1963

Kummerle, R., Grisetti, G., Strasdat, H., Konolige, K., & Burgard, W. (2011). *g2o: A general framework for graph optimization*. In *IEEE international conference on robotics and automation* (pp. 3607–3613).

Latif, Y., Cadena, C., & Neira, J. (2013). Robust loop closing over time for pose graph SLAM. *International Journal of Robotics Research*, *32*(14), 1611–1626.

Leishman, R. C., McLain, T. W., & Beard, R. W. (2014). Relative navigation approach for vision-based aerial GPS-denied navigation. *Journal of Intelligent and Robotic Systems*, *74*(1), 97–111.

Leutenegger, S., Lynen, S., Bosse, M., Siegwart, R., & Furgale, P. (2015). Keyframe-based visual-inertial odometry using nonlinear optimization. *International Journal of Robotics Research*, *34*(3), 314–334.

Long, A., Wolfe, K., Mashner, M., & Chirikjian, G. (2012). The banana distribution is Gaussian: A localization study with exponential coordinates. In *Robotics: science and systems* (p. 8)

Lowry, S., Sunderhauf, N., Newman, P., Leonard, J. J., Cox, D., Corke, P., et al. (2016). Visual place recognition: A survey. *IEEE Transactions on Robotics*, *32*(1), 1–19.

Martinelli, A. (2012). Vision and IMU data fusion: Closed-form solutions for attitude, speed, absolute scale, and bias determination. *IEEE Transactions on Robotics*, *28*(1), 44–60.

Michael, N., Mellinger, D., Lindsey, Q., & Kumar, V. (2010). The GRASP multiple micro-UAV testbed. *IEEE Robotics and Automation Magazine*, *17*(3), 56–65.

Moore, D., Huang, A., Walter, M., Olson, E., Fletcher, L., Leonard, J., & Teller, S. (2009). Simultaneous local and global state estimation for robotic navigation. In: IEEE international conference on robotics and automation (pp. 3794–3799).

Nister, D., & Stewenius, H. (2006). Scalable recognition with a vocabulary tree. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2*, 2161–2168.

Olson, E., & Agarwal, P. (2013). Inference on networks of mixtures for robust robot mapping. *International Journal of Robotics Research*, *32*(7), 826–840.

Olson, E., Leonard, J., & Teller, S. (2006). Fast iterative alignment of pose graphs with poor initial estimates. In *IEEE international conference on robotics and automation* (pp. 2262–2269)

PricewaterhouseCoopers. (2016). Clarity from above: PwC global report on commercial applications of drone technology. https://www.pwc.pl/pl/pdf/clarity-from-above-pwc.pdf. Accessed March 30, 2017.

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., & Ng, A. Y. (2009). ROS: An open-source robot operating system. In: *ICRA workshop on open source software*.

Raffo, G. V., Ortega, M. G., & Rubio, F. R. (2010). An integral predictive/nonlinear H-infinity control structure for a quadrotor helicopter. *Automatica*, *46*(1), 29–39.

Rehder, J., Gupta, K., Nuske, S., & Singh, S. (2012). Global pose estimation with limited GPS and long range visual odometry. In: IEEE international conference on robotics and automation (pp. 627–633)

Roumeliotis, S.I., & Burdick, J. (2002). Stochastic cloning: a generalized framework for processing relative state measurements. In *IEEE International Conference on Robotics and Automation* (pp. 1788–1795)

Scaramuzza, D., & Fraundorfer, F. (2011). Visual odometry: Part I: The first 30 years and fundamentals. *IEEE Robotics and Automation Magazine*, *18*(4), 80–92.

Scaramuzza, D., Achtelik, M. C., Doitsidis, L., Friedrich, F., Kosmatopoulos, E., Martinelli, A., et al. (2014). Vision-controlled micro flying robots: From system design to autonomous navigation and mapping in GPS-denied environments. *IEEE Robotics and Automation Magazine*, *21*(3), 26–40.

Scherer, S., Rehder, J., Achar, S., Cover, H., Chambers, A., Nuske, S., et al. (2012). River mapping from a flying robot: State estimation, river detection, and obstacle mapping. *Autonomous Robots*, *33*(1), 189–214.

Scherer, S., Yang, S., & Zell, A. (2015). DCTAM: Drift-corrected tracking and mapping for autonomous micro aerial vehicles. In *International conference on unmanned aircraft systems* (pp. 1094–1101).

Shen, S., Mulgaonkar, Y., Michael, N., & Kumar, V. (2014). Multi-sensor fusion for robust autonomous flight in indoor and outdoor environments with a rotorcraft MAV. In *IEEE international conference on robotics and automation* (pp 4974–4981).

Sibley, D., Mei, C., Reid, I.D., & Newman, P. (2009). Adaptive relative bundle adjustment. In *Robotics: Science and Systems*

Sivic, Zisserman. (2003). Video Google: A text retrieval approach to object matching in videos. *IEEE International Conference on Computer Vision*, *2*, 1470–1477.

Sunderhauf, N., & Protzel, P. (2013). Switchable constraints vs. max-mixture models vs. RRR—a comparison of three approaches to robust pose graph SLAM. In *IEEE international conference on robotics and automation* (pp. 5198–5203)

Thrun, S., Burgard, W., & Fox, D. (2005). *Probabilistic robotics*. Cambridge: MIT Press.

Tomic, T., Schmid, K., Lutz, P., Domel, A., Kassecker, M., Mair, E., et al. (2012). Toward a fully autonomous UAV: Research platform for indoor and outdoor urban search and rescue. *IEEE Robotics and Automation Magazine*, *19*(3), 46–56.

Weiss, S., & Siegwart, R. (2011). Real-time metric state estimation for modular vision-inertial systems. In *IEEE international conference on robotics and automation* (pp. 4531–4537).

Weiss, S., Achtelik, M. W., Lynen, S., Chli, M., & Siegwart, R. (2012). Real-time onboard visual-inertial state estimation and self-calibration of MAVs in unknown environments. In *IEEE International Conference on Robotics and Automation* (pp. 957–964).

Wheeler, D. O., Nyholm, P. W., Koch, D. P., Ellingson, G. J., McLain, T. W., & Beard, R. W. (2016). Relative navigation in GPS-degraded environments. In *Encyclopedia of aerospace engineering* (pp. 1–10). Hoboken: Wiley

Wheeler, D. O., Koch, D. P., Jackson, J. S., Mclain, T. W., & Beard, R. W. (2018). Relative navigation: A keyframe-based approach for observable GPS-degraded navigation. *IEEE Control Systems Magazine*, *38*(4), 30–48.

Zhang, J., Kaess, M., & Singh, S. (2014). Real-time depth enhanced monocular odometry. In *IEEE/RSJ international conference on intelligent robots and systems* (pp. 4973–4980).

David O. Wheeler received the Ph.D. degree in electrical engineering from Brigham Young University (BYU) in 2017, and the B.S. degree from BYU in mechanical engineering in 2013 graduating magna cum laude. He is currently an autonomy engineer at Uber Advanced Technology Group.

Daniel P. Koch is a Ph.D. student in the Mechanical Engineering Department at Brigham Young University, and a National Defense Science and Engineering (NDSEG) fellow. He received the B.S. degree in mechanical engineering, summa cum laude, with a minor in computer science from Brigham Young University in 2014.

James S. Jackson is a Ph.D. student in the Mechanical Engineering Department at Brigham Young University. He received the B.S. degree in mechanical engineering from Brigham Young University in 2014.

Gary J. Ellingson is a Ph.D. candidate in the Mechanical Engineering Department at Brigham Young University. He completed the B.S. degree, with university honors and magna cum laude, in Mechanical Engineering with a minor in computer science at Brigham Young University in 2014.

Paul W. Nyholm received the B.S. and M.S. degrees in mechanical engineering from Brigham Young University in 2013 and 2015. He currently works at Lawrence Livermore National Laboratory.

Timothy W. McLain is a professor in the Department of Mechanical Engineering at Brigham Young University (BYU). He received the B.S. and M.S. degrees in mechanical engineering from BYU. While completing his Ph.D. work at Stanford University, he worked with the Monterey Bay Aquarium Research Institute on the control of underwater robotic vehicles. He joined BYU in 1995. During the summers of 1999 and 2000, he was a visiting scientist at the Air Force Research Laboratory, where he initiated research on unmanned aircraft, an area on which he continues to focus. With Randal Beard, he is the author of the textbook Small Unmanned Aircraft: Theory and Practice (Princeton University Press, 2012). He is currently the director of the Center for Unmanned Aircraft Systems under the National Science Foundation Industry/University Cooperative Research Center program.

Randal W. Beard received the B.S. degree in electrical engineering from the University of Utah, Salt Lake City, in 1991, the M.S. degree in electrical engineering in 1993, the M.S. degree in mathematics in 1994, and the Ph.D. degree in electrical engineering in 1995, all from Rensselaer Polytechnic Institute, Troy, New York. Since 1996, he has been with the Electrical and Computer Engineering Department at Brigham Young University, Provo, Utah, where he is currently a professor. In 1997 and 1998, he was a Summer faculty fellow at the Jet Propulsion Laboratory, California Institute of Technology, Pasadena. In 2006 and 2007 he was a visiting research fellow at the Air Force Research Laboratory, Munitions Directorate, Eglin Air Force Base, Florida. His primary research focus is autonomous control of miniature air vehicles and multi-vehicle coordination and control. He is a past associate editor for IEEE Control Systems Magazine, the Journal of Intelligent and Robotic Systems, and the IEEE Transactions on Automatic Control. He is a fellow of IEEE.