Accepted Manuscript
Version of record at: https://doi.org/10.1142/S2737480722500200

Resilient Ground Vehicle Autonomous Navigation in GPS-denied Environments

Kleio Baxevani*, Indrajeet Yadav†, Yulin Yang‡, Michael Sebok§, Herbert G. Tanner¶ and Guoquan Huang‖

Center for Autonomous & Robotic Systems,
University of Delaware,
Newark, DE, 19716, USA

*kleiobax@udel.edu
†indragt@udel.edu
‡yuyang@udel.edu
§sebokm@udel.edu
¶btanner@udel.edu
‖ghuang@udel.edu

Co-design and integration of vehicle navigation and control and state estimation is key for enabling field deployment of mobile robots in GPS-denied cluttered environments, and sensor calibration is critical for successful operation of both subsystems. This paper demonstrates the potential of this co-design approach with field tests of the integration of a reactive receding horizon-based motion planner and controller with an inertial aided multi-sensor calibration scheme. The reported method provides accurate calibration parameters that improve the per-formance of the state estimator, and enable the motion controller to generate smooth and continuous minimal-jerk trajectories based on local LiDAR data. Numerical simulations in Unity, and real-world experimental results from the field corroborate the claims of efficacy for the reported autonomous navigation computational pipeline.

Keywords: Field robots; calibration and identification; motion and path planning.

1. Introduction

1.1. *Motivation*

Autonomous vehicles continue to expand their domain of application on the ground, in the air, on the surface of the sea, and underwater. While the underwater space presents clear cases of robot deployment unmapped, uncertain, and GPS-denied environments, there are also multiple terrestrial cases too, including but not limited

* Corresponding author.

Accepted Manuscript
Version of record at: https://doi.org/10.1142/S2737480722500200

to indoor or subterranean navigation, as well as navigation in forest environments under a thick canopy.

To carry out navigation tasks in such a wide range of environments, robots utilize a heterogeneous suite of sensors, including inertial measurement unit (IMU), multiple cameras, range sensors (e.g. LiDARs), in addition to standard odometry sensors. Still, the amount of uncertainty and intensity of environmental disturbances — especially in unstructured and GPS-denied environments — presents significant remaining challenges to autonomous navigation. In GPS-denied environments, the perception and localization system is further challenged by the compounding effect of LiDAR error, IMU attitude error, and boresight alignment error.[1] Accuracy then can be dropped to levels which cannot be corrected due to the unavailability of GPS information, and uncertainty propagation can lead to catastrophic effects.

Thus, a key attribute for successful autonomous navigation in such environments is persistent, resilient, and accurate estimation of the position and velocity of the robot that includes observation of unknown environmental structures, is combined with a robust motion planning and control methodology that adapts in real-time to measurements.

Navigation in GPS-denied environments is always possible via semi-autonomous control schemes,[2] which naturally minimize the reliance on accurate and consistent sensor fusion of data from LiDAR, GPS, and other sensitive sensors. The reliance on human intervention, however, is not always desirable; in fact, there can be instances where such interventions can introduce additional risk,[3,4] further motivating schemes that allow for full vehicle autonomy.

Such autonomy relies heavily on the accuracy of the sensor information. It has been shown that this accuracy can be significantly improved by multi-sensor calibrating techniques[1,5] which exploit high-rate inertial navigation and can handle a multitude of asynchronous sensors commonly found on robots (e.g. IMU, cameras, LiDARs, and wheel odometry) while requiring only initial calibration guesses.

Achieving long-term autonomy through real-time perception and adaptive trajectory generation and control is a significant challenge for robots deployed in the field and under real-world conditions, especially when they operate in outdoor and sometimes GPS-denied environments. Operation in such environments requires the robots to remain robust, resilient, and adaptive in real-time.[6]

Resilient autonomy can be built on two methodological pillars: robust perception and estimation that incorporated automatic sensor calibration, and adaptive motion planning and control. Having reported on both of these component technologies in isolation earlier,[5,7] this paper elaborates more on their methodological integration. The goal is to realize a robust and reactive motion planning and control architecture appropriate for navigation in unknown and GPS-denied environments involving real-time, fast, sensor-based reactive trajectory generation, coupled with receding horizon motion planning and control (Fig. 2).

Accepted Manuscript
Version of record at: https://doi.org/10.1142/S2737480722500200

1.2. *Related literature*

Inertial navigation systems (INSs) that have been widely used for providing six degrees of freedom (DOF) pose estimation when navigating in 3D space. However, due to the noises and biases that corrupt the IMU readings, simple integration of the local angular velocity and linear acceleration measurements can cause large drifts in a short period of time, in particular, when using cheap MEMS IMUs. To mitigate this issue, additional sensors (e.g. camera, LiDAR, and underwater sonar) are often used, i.e. aided INS. Among possible exteroceptive sensors, optical cameras which are low-cost and energy-efficient while providing rich environmental information are ideal aiding sources for INS and thus, visual-inertial navigation system (i.e. VINS) has recently prevailed, in particular, when navigating in GPS-denied environments (e.g. indoors).[8] While many different VINS algorithms were developed in the last decade, the extended Kalman filter (EKF)-based methods are still among the most popular ones.[9]

There are two types of methods for real-time sensor-based robot trajectory generation: (a) graph-search methods,[10] which can generate globally optimal solutions; and (b) parametric optimization methods, which search the continuum of solutions to find locally optimal solutions.[11–13] The main difference between the two relates to the type of information the system possesses. If the system operates in a known environment, then a global motion planner can generate the complete trajectory from start to destination. If prior information about the robot's workspace is not available or the vehicle operates in a completely unknown environment, then local planners can generate trajectory segments based on local sensory information, e.g. the features identified in the field of view (FOV) of the vehicle. The quality and utility of such local trajectories can significantly increase, contributing to robustness in field robotics applications, by directly incorporating terrain features[14]; in many real-world scenarios, however, such data are either not available or not particularly accurate and reliable since terrain properties can often change based on weather conditions and seasonal effects. Ultimately, the ability of a local ground vehicle planner to navigate safely and robustly is facilitated by minimal reliance on prior information, frugal utilization of sensory data, and respect of the vehicle's kinematic and dynamic constraints.[15,16]

A proven approach to robot navigation through online reference trajectory generation is based on the real-time generation of a set of *waypoint* poses (keyframes). These waypoints are then sequentially linked with the use of continuous curves parametrized as time polynomials. In applications of unknown environment exploration, such approaches are usually referred to as *viewpoint* planning.[17] This process involves the construction of a 3D workspace representation, within which the sequence of viewpoints is followed. To evaluate the quality of waypoints as a means of directing the robot toward its goal, work in literature attempts to quantify the amount of unexplored information that the vehicle is expected to gather from each pose.[18] Even though these type of metrics can be used to improve the performance

Accepted Manuscript
Version of record at: https://doi.org/10.1142/S2737480722500200

of the state estimator running on the vehicle, they still cannot offer any type of guarantees regarding the energy that would be required by the vehicle to track the generated trajectories; given that power autonomy can be crucial in field deployments, the latter feature can rise high in the priority list of robot mission specifications.

As waypoints are defined in real-time based on sensor information, there is a need for a method to generate trajectories that interpolate between these waypoints in a way compatible with the robot dynamics. One reliable approach, which also allows for on-line adaptation of said trajectories, is a receding horizon strategy. The implementation of such a receding horizon planner would involve the reactive sampling of the set of *safe* trajectories within the FOV of the vehicle's sensors, thereby decoupling local collision avoidance from global guidance toward the navigation goal.[19] Alternatives within this same general category can be found in approaches that involve reactively generating reference trajectories in the Frenet frame of the vehicle for the purpose of lane preservation.[20] In the former case, in which both local and global motion planners are employed, the failure of the global planner has a detrimental effect on the vehicle's navigation performance; in the latter case, performance is typically acceptable in structured and known environments (e.g. within a street network) but often suffers in off-road unstructured outdoor environments.

1.3. *Contributions and organization*

The key contribution of this paper is the *realization and testing of a reactive receding horizon motion planner and controller capable of operating in unknown GPS-denied environments, within a computational pipeline that incorporates automatic calibration of the vehicle sensors that inform the robot's (VINS).* Most specifically

- it is applicable to cases with both static and moving goal configurations;
- it generates trajectories in a new way by efficiently solving a multi-objective optimal control problem;
- it is experimentally tested with speeds of up to 4.5–5 m/s in a GPS-denied off-road area; and
- it minimizes the estimation error by accurate off-line sensor calibration leading to more accurate navigation.

The integrated state estimation, motion planning, and control architecture can autonomously steer a ground robot along safe and dynamically feasible trajectories that locally minimize motion jerk — the latter taken as a proxy metric of energy efficiency. The methodology does not rely on prior maps, externally provided localization information (GPS), or global planners, neither does it force the vehicle along some "lane-following" reference trajectory, and has shown to work reasonably well in off-road conditions (Fig. 1). The local motion planner evaluates potential trajectory waypoints based on their proximity to the obstacles and to the final robot navigation

Accepted Manuscript
Version of record at: https://doi.org/10.1142/S2737480722500200

Fig. 1. A Clearpath Robotics Jackal robot moving autonomously along a trail within a wooded outdoor space. Under a thick canopy in such environments autonomous robots should be able to localize and navigate without relying on GPS. (The Ethernet cable visible was utilized solely for remote data collection and telemetry.)

goal, as well as their divergence from the previously generated trajectory. This planner (i) generates a set of feasible rays within the vehicle sensors FOV, (ii) utilizes a 2D costmap of the environment created from range sensors to check for the prospect of local collisions along these rays, (iii) subsequently discards rays that appear to intersect with obstacles, (iv) selects the one that optimizes a heuristic objective function that balances (collision) safety against convergence speed (see Sec. 3.2), and then (v) generates a dynamically feasible reference trajectory along the direction of the optimal ray. This process is repeated in a receding horizon manner, i.e. the controller tracks a small segment of the reference trajectory and before this tracking is finished, a new segment is appended (in a dynamically feasible manner) to the old trajectory and the controller switches to the new segment once the old is completed (cf. Ref. 21).

The rest of this paper is organized as follows: Sec. 2 provides a brief overview of the system. Following that, Sec. 3 lays out the technical and mathematical description of the system. Section 4 presents simulation and experimental results from Unity and real-world outdoor testing, respectively, which demonstrate the efficacy of the calibration and motion planning modules of the autonomy architecture on the ground vehicle. This paper closes with a quick overview and future work in Sec. 5.

2. Navigation Architecture Overview

The block diagram of Fig. 2 illustrates the architecture of the overall calibration and motion planning system. The arrows indicate the information and data flow between the different system modules.

The offline calibration module provides spatial-temporal calibration parameters to the state estimator to improve accuracy. Datasets with AuRco tags[22] are first

Accepted Manuscript
Version of record at: https://doi.org/10.1142/S2737480722500200

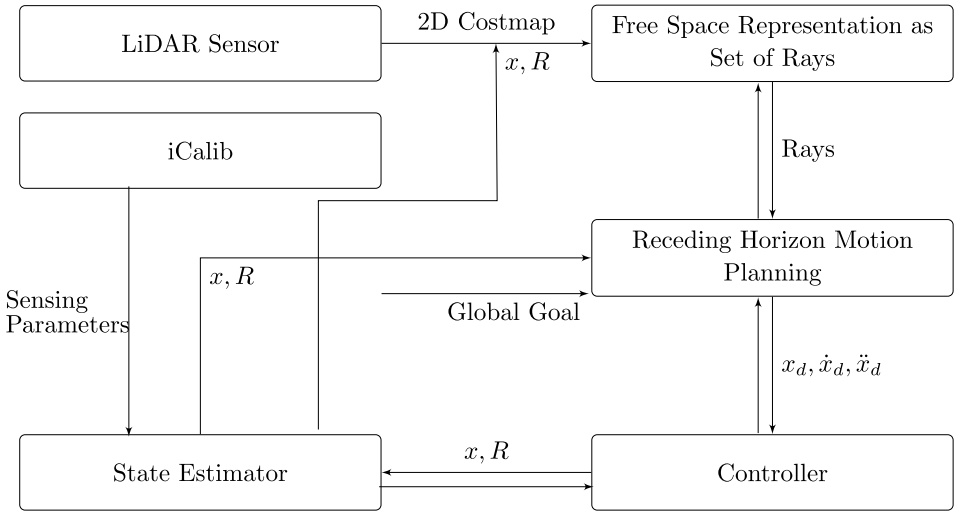

Fig. 2. Block diagram of the motion planning and control architecture.

collected, using all the needed sensors (including at least one IMU and one camera as a base sensor pair) on the robot (Fig. 5). A batch least-squares problem is formulated based on a base sensor trajectory. Information from other sensor measurements (auxiliary camera measurements to tags, LiDAR plane measurements[23] and wheel encoder readings[24]), which encapsulate the spatial-temporal calibration to the base IMU constraints, is captured in additional cost elements within the batch least-squares optimization process.

For navigation, a 2D costmap is constructed based on LiDAR data. Emanating from the location of the sensor and covering its FOV, a set of rays is generated which extends over unoccupied map cells. The receding horizon planning module then (i) selects an (intermediate) obstacle-free point that corresponds to the spatial point on the edge of the FOV that is closest to the navigation goal, (ii) assigns a cost to each ray as an equally weighted sum of the robot's proximity to the obstacle, the proximity to the intermediate point, and the angle of the ray relative to the robot's yaw angle, (iii) selects the safest path to the intermediate point, and finally (iv) constructs the feasible minimum-jerk trajectory along this path. The block diagram of the process is presented in Fig. 3.

An initial segment of that minimum-jerk reference trajectory is then provided to a differential-geometric trajectory tracking controller, that utilizes a unicycle model to steer the ground robot along its way to the local goal point within the FOV. In a typical receding horizon fashion, before the end of that initial segment is reached, the vehicle uses the latest LiDAR information to update the local costmap, generate new trajectory, and transitions smoothly between reference trajectory segments. The replanning and trajectory tracking process are repeated until the final static global destination is reached.

Accepted Manuscript
Version of record at: https://doi.org/10.1142/S2737480722500200

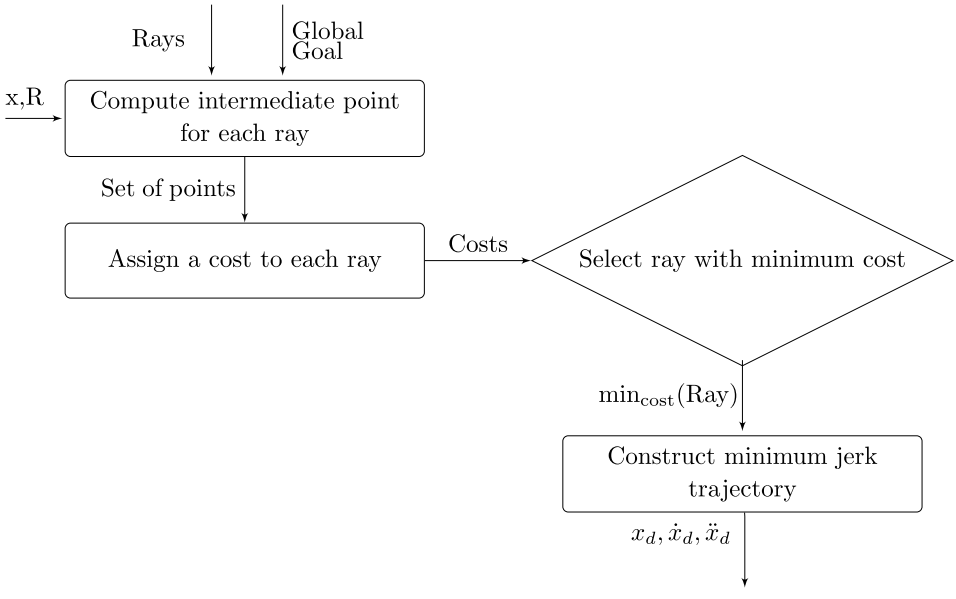

Fig. 3. Block diagram of the receding horizon planning module.

3. Implementation

3.1. *Inertial-aided multi-sensor calibration*

This section describes the offline inertial aided multi-sensor calibration algorithm (iCalib).[5] It is based on a nonlinear least-squares multi-sensor calibration system, which leverages high-rate inertial navigation and can handle a multitude of asynchronous sensors commonly found on robots (e.g. IMU, cameras, LiDARs, and wheel odometry) while requiring only initial calibration and initial estimates. With $\mathbf{h}_S$ denoting the measurement model of sensor S, and $\mathbf{n}_S$ being a vector of white Gaussian noise with covariance $\mathbf{R}_S$, the measurement of sensor S is expressed as

$$\mathbf{z}_S = \mathbf{h}_S(\mathbf{x}) + \mathbf{n}_S, \quad \mathbf{n}_S \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_S).$$

A nonlinear least-squares problem can now be formulated:

$$\min_{\mathbf{x}} \sum \|\mathbf{z}_S - \mathbf{h}_S(\mathbf{x})\|^2_{\mathbf{R}_S^{-1}} \tag{1}$$

requiring an initial guess $\hat{\mathbf{x}}$ to start the optimization process. By incorporating multiple sensor measurement costs into the formulation, the above least-squares cost function (1) can be augmented to

$$\min_{\mathbf{x}} \sum \mathbb{C}_I + \sum \mathbb{C}_C + \sum \mathbb{C}_L + \sum \mathbb{C}_W, \tag{2}$$

where $\mathbb{C}_I$, $\mathbb{C}_C$, $\mathbb{C}_L$, and $\mathbb{C}_W$ represent the measurement costs from IMU, camera, LiDAR, and wheel encoder, respectively (see Ref. 5 for their explicit definition), the

Accepted Manuscript
Version of record at: https://doi.org/10.1142/S2737480722500200

least-squares problem maps to maximum likelihood estimation (MLE), which is solved via iterative algorithms. Compared to other existing calibration methods (Table 1), iCalib relies on sensor motion to gain calibration constraints and can handle both spatial-temporal calibration among all the listed sensors.

Table 1. Existing multi-sensor calibration algorithms. Parameter t_{off} refers to time offset calibration between sensors.

Algorithm	IMU	Camera	LiDAR	Wheel	t_{off}	Target
iCalib[5]	×	×	×	×	×	×
LI-Calib[25]	×		×			
MIMC[26]	×	×			×	
LIC[23]	×	×	×		×	
VIWO[24]	×	×		×	×	

3.2. *Reactive planning and control*

This section summarizes the planning and control stack implemented on the robot. The planning part utilizes a modified version of our aerial vehicle receding horizon planner[15] tailored to generate minimum jerk trajectories using obstacle information from a 2D local costmap. The trajectory generation is reactive in the sense that it continuously takes environment information via the costmap, replans the trajectories, and tracks them in a receding horizon fashion.

3.2.1. *Planner*

A typical 2D costmap for a simulated off-road environment is shown in Fig. 4. It encodes the pointcloud information from the LiDAR into three broad cell categories: free, occupied, and unknown. The goal of the planner is to generate safe trajectories that are completely contained within the free cells. The categorization of occupied cells is further refined into "lethal" (in obstacle) "inscribed" (too close) and "circumscribed" (precarious). These types of cells are shown by different colors

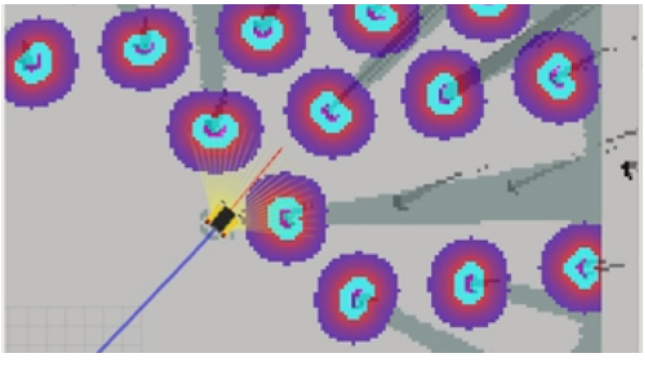

Fig. 4. A typical 2D costmap generated in a simulated indoor environment.

Accepted Manuscript
Version of record at: https://doi.org/10.1142/S2737480722500200

around the obstacle in Fig. 4. The planner ensures that the center of the robot does not fall within any of these three types of cells.

The planner then generates an ensemble of rays in a pre-selected visibility cone of the robot. Although the LiDAR can have a full 3D view around the ground vehicle, trajectories extending directly behind the robot are usually not dynamically feasible. A feasible *visibility cone* is therefore defined, which in our case is an area spanning a $\pm 60^\circ$ angle around the forward heading direction. An ensemble of rays is then generated *within* this visibility cone and rays that fall within either lethal, inscribed or circumscribed cells of the pointcloud are discarded. The ray generation and collision checking technique is based on earlier work[15]; the difference here is that the obstacle information now comes from the costmap.

A linear combination of three components is used to assign a cost to each free ray. Let p be the total number of free rays, let d_i denote the distance of end point of the ith ray to the final goal point, $c_{\text{collision}_i}$ be the cost of nearest obstacle from the ith ray, and $c_{\text{divergence}_i}$ denote the divergence cost of ith ray from the previously generated trajectory. Then, for the positive scalar weights $k_1, k_2, k_3 \in (0, 1)$, the total cost associated with ray i expressed for $i \in [0, \ldots, p]$ is

$$c_i = k_1 \frac{d_i}{d_{\max}} + k_2\, c_{\text{collision}_i} + k_3 c_{\text{divergence}_i},$$

where the divergence cost compares the angle of each free ray in the ensemble from the tangent to the trajectory generated in the last planning cycle, and all cost components are normalized in the $(0, 1)$ range. Taking the best ray, and in each planning cycle, the receding horizon planner generates a dynamically feasible minimum jerk reference trajectory of time duration T (referred to as the *planning horizon*) along this direction, using the user-specified desired nominal speed of the vehicle.[15] The robot utilizes a fairly standard non-holonomic output tracking controller (see below) to track an initial portion of this reference trajectory for a small time interval (referred to as the *control horizon* — a fraction of the planning horizon T). The length of the control horizon is dependent on the vehicle's speed, LiDAR's update rate, and computational capabilities onboard the vehicle. Generally, the control horizon should be longer than the update rate of the LiDAR plus some safety margin.

Upon receiving new pointcloud data, the robot modifies the local costmap, generates a new trajectory and appends the newly computed reference trajectory segment to the end of the trajectory segment it is currently implementing. When the robot reaches the end of the control horizon of the reference trajectory segment currently being implemented, it smoothly switches to the next trajectory. The process between sensor updates constitutes a *replanning cycle*.

3.2.2. *Controller*

Let (x, y) be the current position of the robot, v_x and v_y the two components of its velocity vector v, and θ and ω its speed, current heading and angular velocity, respectively; then, assuming unicycle kinematics, the ground vehicle's motion can be

Accepted Manuscript
Version of record at: https://doi.org/10.1142/S2737480722500200

represented as follows:

$$\dot{x} = v \cdot \cos\theta, \dot{y} = v \cdot \sin\theta, \quad \dot{\theta} = \omega. \tag{3}$$

Denote $[x_r(t), y_r(t)]$ the reference trajectory to be tracked, and define $\theta_r(t) \triangleq \arctan(\frac{\dot{y}_r(t)}{\dot{x}_r(t)})$. Then

$$\dot{\theta}_r(t) = \frac{\ddot{y}_r(t)\dot{x}_r(t) - \ddot{x}_r(t)\dot{y}_r(t)}{\dot{x}_r^2(t) + \dot{y}_r^2(t)}.$$

With this, the current configuration of the vehicle can be expressed as $[x, y, \theta]$. For a small positive constant ϵ, define

$$z_1 = x + \epsilon\cos\theta, \quad z_2 = y + \epsilon\sin\theta.$$

Similarly for the reference motion, one defines

$$z_{r1}(t) = x_r(t) + \epsilon\cos\theta_r(t), \quad z_{r2}(t) = y_r(t) + \epsilon\sin\theta_r(t)$$

and expresses their derivatives as

$$\dot{z}_{r1}(t) = \dot{x}_r(t) - \epsilon\dot{\theta}_r(t)\sin\theta_r(t)\dot{z}_{r2}(t) = \dot{y}_r(t) + \epsilon\dot{\theta}_r(t)\cos\theta_r(t).$$

Then for $k > 0$, the control law

$$u_1(t) = \dot{z}_{r1}(t) - k(z_1(t) - z_{r1}(t)), \quad u_2(t) = \dot{z}_{r2}(t) - k(z_2(t) - z_{r2}(t))$$

on the integrator system $\dot{z}_1(t) = u_1(t)$, $\dot{z}_2(t) = u_2(t)$, renders the $z_r - z$ output error dynamics exponentially stable. Mapping back to the original unicycle coordinates yields the robot's control input:

$$v(t) = u_1(t)\,\cos\theta(t) + u_2(t)\,\sin\theta(t), \quad \omega(t) = -u_1(t)\frac{\sin\theta(t)}{\epsilon} + u_2(t)\frac{\cos\theta(t)}{\epsilon}.$$

Fig. 5. Sensor setup for Jackal containing Velodyne VLP-16 LiDAR, Microstrain IMU GX5-25, Realsense T265 cameras, and FLIR Blackfly camera. Jackal itself contains a wheel encoder.

Accepted Manuscript
Version of record at: https://doi.org/10.1142/S2737480722500200

3.3. *Hardware*

The mobile robot base is a Jackal platform by Clearpath[a] and the sensor payload includes a Velodyne VLP-16 LiDAR, a microstrain IMU GX5-25, two Realsense cameras (T265 and D435i), a FLIR Blackfly camera and a wheel encoder. The LiDAR device is rigidly mounted on top of the robot and the Realsense cameras along with the Blackfly camera are mounted below the LiDAR to provide fine pointcloud and depth information. This configuration is depicted in Fig. 5.

4. Numerical and Experimental Validation

The validation of the navigation stack was carried out using both simulated obstacle-dense environments as well as outdoor field testing. Simulations were conducted in a Unity simulation environment with test environments that were modeled in Blender (Fig. 6). The parameters of the robotic vehicle in simulation were set to mimic the behavior of the Clearpath Jackal used in the real-world tests. Rviz was utilized to visualize the costmap data and generate plots of the vehicular motion. A similar approach was utilized for the testing of the robot in the field test with the testing environment and parameters selected based on their general similarity to the simulation environment.

4.1. *Calibration testing*

The data to evaluate the automatic calibration module were collected using the robot in the environment of Fig. 5. A ramp was utilized in order to reduce the degeneracy between the IMU and other sensors such as the camera, the wheel encoders, and the LiDAR. To minimize the interpolation errors of the base sensor, the initial time offset guesses were set to zero. Two scenarios were designed to evaluate the performance of iCalib: IMU+Camera+LiDAR with MSG-CAL[27] as the reference, and IMU

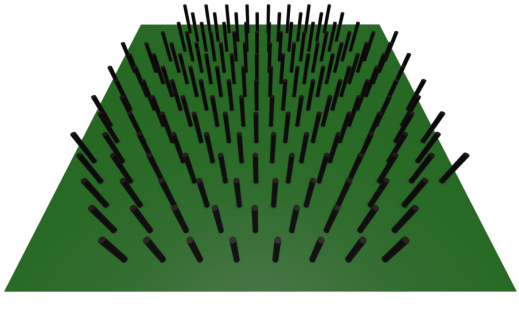

Fig. 6. The high density environment used for Jackal simulations.

[a] https://clearpathrobotics.com/.

Accepted Manuscript
Version of record at: https://doi.org/10.1142/S2737480722500200

+Camera+Wheel with manually measured reference values. Four datasets were drawn from MSG-CAL; their average was used as reference.

To allow direct comparisons, it is necessary to map the proposed system's converged CAM-IMU and LiDAR-IMU transformations to a relative CAM-LiDAR calibration, before computing the orientation quaternion error relative to a hand measured datum (Fig. 7). Here is a difference of less than 0.5° between the orientation errors and a difference of 1 cm between the translation with the time offsets converging to stable values.[5]

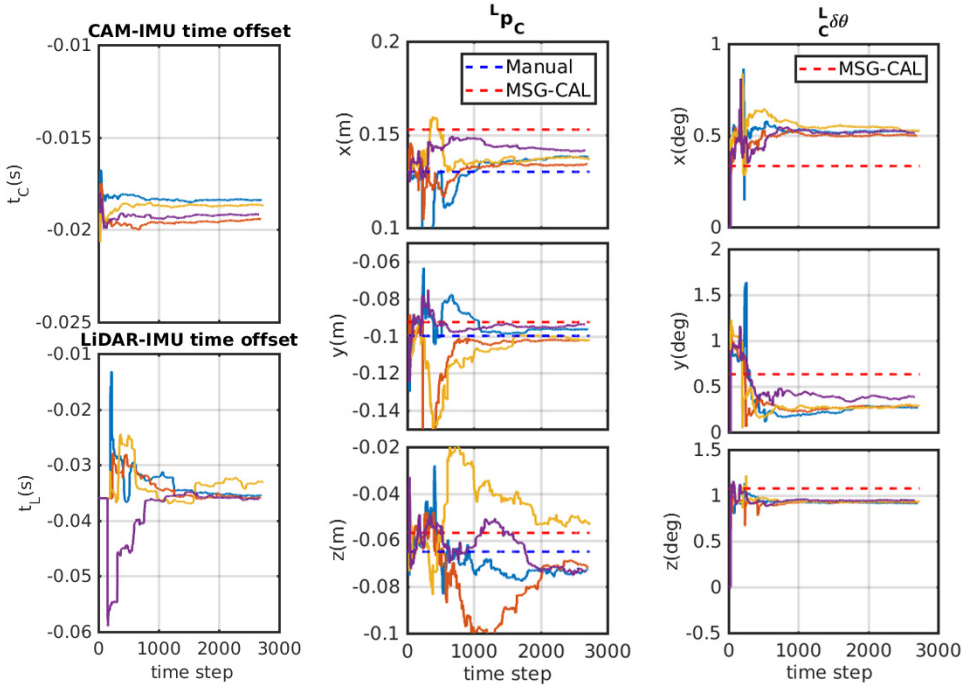

Fig. 7. Spatial-temporal calibration for IMU, Camera, and LiDAR. We transferred the IMU-Cam and IMU-LiDAR calibration to Cam-LiDAR calibration for comparison with MSG-CAL, which is represented by the red dotted line.[5]

Figure 8 indicates convergence for the time offset parameter of the base camera to a constant value. The figure however demonstrates lack of convergence for the variables for roll, pitch, and z translation of the IMU-wheel calibration set, an outcome that may be attributed to loss of observability. One potential culprit is the utilization of the 2D wheel odometry model for 3D rigid body calibration. Calibration of the wheel odometry extrinsics without any cross-sensor constraints is currently still an open problem.

Accepted Manuscript
Version of record at: https://doi.org/10.1142/S2737480722500200

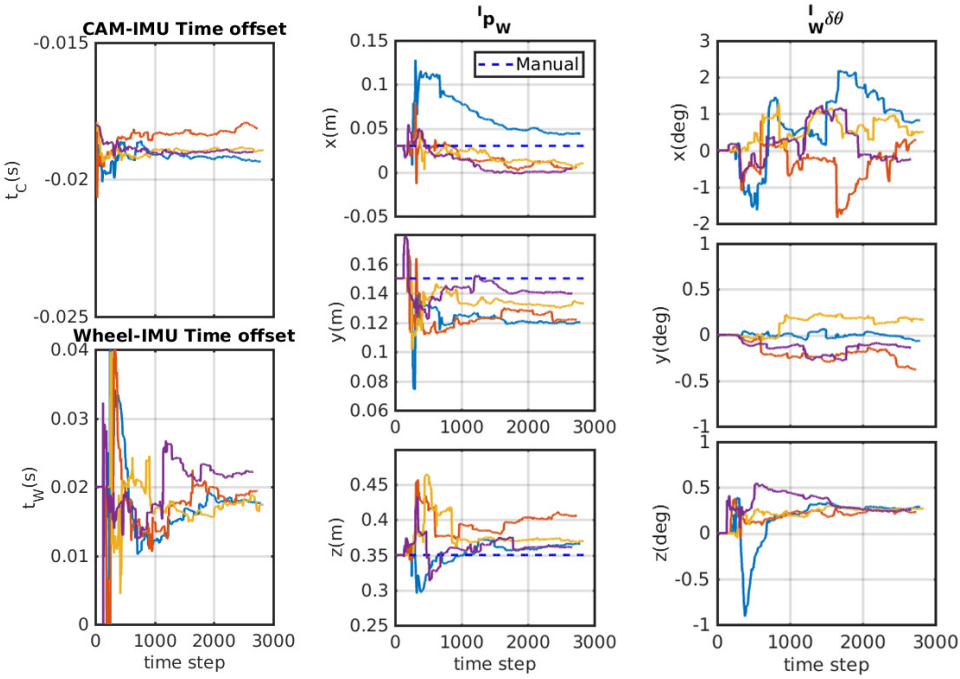

Fig. 8. Spatial-temporal calibration for IMU, camera, and wheel encoder. Due to limited space, only the spatial calibration between IMU and wheel encoder is shown. The IMU-wheel calibration converges slowly mainly due to the degeneracy of the vehicle's planar motion.[5]

4.2. *Numerical testing for navigation and control*

A numerical model of Jackal was tested in simulated environments with low, medium, and high obstacle densities. Each environment contained cylindrical obstacles of 1 m in diameter with varying center-to-center distances based on the desired obstacle density. Obstacle configuration was selected to mimic the distribution of trees in the wooded outdoor testing area utilized, and the increase in density is intended to challenge the ability of the algorithm to navigate highly cluttered environments. As expected, the high-density configuration, featuring a center-to-center distance between obstacles of $\sim$5.5 m, presents the most challenge. This obstacle density led the planner to generate tightly curved trajectories in order to avoid the obstacles.

The path that the robot carves through the high-density simulated forest environment is marked in Fig. 9. The robot's goal location in this particular run was set at $\{40, 40\,\mathrm{m}\}$. Note that the robot generates a path that does not deviate considerably from the straight-line path between the starting position and the goal. The generated motion of the vehicle is smooth and continuous as expected due to planner's jerk minimization.

Accepted Manuscript
Version of record at: https://doi.org/10.1142/S2737480722500200

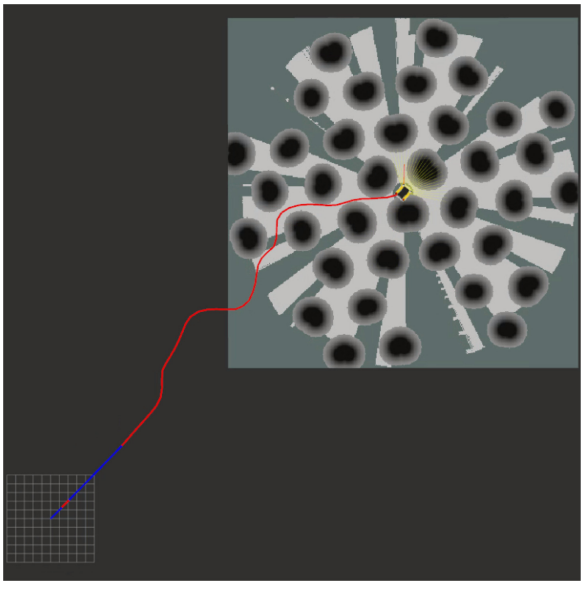

Fig. 9. The simulated trajectory and costmap for the Jackal in the high density testing environment.

4.3. *Outdoor experimentation results*

For real-world testing and system validation, experiments were run with the robot shown in Fig. 5, deployed in a wooded environment that resembles the one used in simulation studies. Outdoor testing was conducted without any predefined map of the area, and the costmaps used were generated in real-time using only LiDAR data. Figure 10 shows the costmap generated in one of the wooded environments utilized

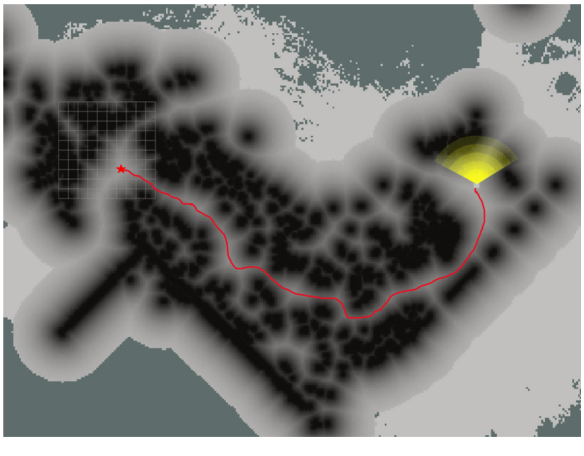

Fig. 10. The final Jackal position and costmap generated in the outdoor wooded environment. Black regions represent obstacles (trees or fence structures).

Accepted Manuscript
Version of record at: https://doi.org/10.1142/S2737480722500200

for testing, after the robot has reached its final destination. The robot begins at the starting position in the center of the grid shown on the left and navigates through an off-road forested environment to reach its goal at $\{40, 0\,\mathrm{m}\}$. Note that there are a few narrow paths through the woods that could potentially lead the robot to its destination, but there was no prior information on either of these solutions or on the environment's topology. In the testing runs conducted, the robot was consistently able to navigate through the off-road environment and reach the destination location on the other rightmost side of the small patch of woods. The resulting error between the final robot position and desired goal could range in 1–5 m between runs — it should be noted, however, that no GPS information is utilized here, and uncertainty over the initial robot heading also factors in the outdoor localization error. Overall, the motion generated in outdoor testing generally matches the behavior seen in simulation, and both were comparably smooth (minimum jerk as a proxy for power efficiency) relative to the motion optimization criteria used.

5. Conclusions and Future Work

Completely autonomous navigation for unmanned ground vehicles (UGV) in unknown GPS-denied off-road environments can be particularly challenging, requiring careful co-design of sensor calibration, state estimation and motion planning processes. Effective sensor calibration and robust reactive motion planning and control are shown to be effective in such contexts, when tested in unstructured off-road environments without utilizing GPS. An autonomous navigation pipeline along such a direction incorporates a novel integration of two key components: accurate off-line sensor calibration, and reactive receding horizon motion planning and control. Potential future extensions include tightly coupled visual-inertial-wheel odometry with online sensor calibration, which could facilitate effective fusion of monocular images together with IMU and wheel encoder readings, to provide accurate ego-motion tracking.

Acknowledgments

This work was supported by U.S. Army Combat Capabilities Development Command — Army Research Lab via award # W911NF-20-2-0098. Thanks to Dr. Kun Fu for the 3D printing of the sensor support, and to Wenxuan (Owen) Li for involvement in the Jackal experiments.

References

1. W. Liu, Z. Li, S. Sun, R. Malekian, Z. Ma and W. Li, Improving Positioning Accuracy of the Mobile Laser Scanning in GPS-Denied Environments: An Experimental Case Study, *IEEE Sens. J.* **19**(22), 10753–10763 (2019).
2. Y. Endo, J. C. Balloch, A. Grushin, M. W. Lee and D. Handelman, Landmark-Based Robust Navigation for Tactical UGV Control in GPS-Denied Communication-Degraded

Environments, in *Proceedings of the Unmanned Systems Technology XVIII*, Vol. 9837 (Baltimore, Maryland, 2016), pp. 114–126.

3. I. Kostavelis and A. Gasteratos, Robots in Crisis Management: A Survey, in *Proceedings of the Information Systems for Crisis Response and Management in Mediterranean Countries* (Springer International Publishing, 2017), pp. 43–56.
4. K. D. Crowley and J. F. Ahearne, Managing the Environmental Legacy of U.S. Nuclear-Weapons Production: Although the Waste from America's Arms Buildup will Never Be "Cleaned Up," Human and Environmental Risks Can Be Reduced and Managed, *Am. Sci.* **90**(6), 514–523 (2002).
5. Y. Yang, W. Lee, P. Osteen, P. Geneva, X. Zuo and G. Huang, iCalib: Inertial Aided Multi-Sensor Calibration, in *VINS Workshop at IEEE International Conference on Robotics and Automation*, Xi'an China, 2021.
6. S. Eiffert, N. D. Wallace, H. Kong, N. Pirmarzdashti and S. Sukkarieh, Resource and Response Aware Path Planning for Long-Term Autonomy of Ground Robots in Agriculture, arXiv:2105.10690.
7. I. Yadav and H. G. Tanner, Exact Reactive Receding Horizon Motion Planning for Aerial Vehicles, in *Proceedings of the IEEE International Conference on Unmanned Aircraft Systems* (Athens, Greece, 2021), pp. 836–842.
8. T. Qin, P. Li and S. Shen, VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator, *IEEE Trans. Robot.* **34**(4), 1004–1020 (2018).
9. M. Zhang, X. Xu, Y. Chen and M. Li, A Lightweight and Accurate Localization Algorithm Using Multiple Inertial Measurement Units, *IEEE Robot. Autom. Lett.* **5**(2), 1508–1515 (2020).
10. E. Rippel, A. Bar-Gill and N. Shimkin, Fast Graph-Search Algorithms for General-Aviation Flight Trajectory Generation, *J. Guid. Control Dyn.* **28**(4), 801–811 (2005).
11. C. Richter, A. Bry and N. Roy, Polynomial Trajectory Planning for Aggressive Quadrotor Flight in Dense Indoor Environments, in *Robotics Research* (Springer International Publishing, 2016), pp. 649–666.
12. D. Mellinger and V. Kumar, Minimum Snap Trajectory Generation and Control for Quadrotors, in *Proceedings of the IEEE International Conference on Robotics and Automation* (Shanghai, China, 2011), pp. 2520–2525.
13. Z. Wang, Q. Zong, H. Shen, B. Tian and H. Lu, A Parallel Mapping and Planning Strategy for Micro Aerial Vehicle, *Guid. Navig. Control* **2**(02), 2250013 (2022).
14. T. M. Howard and A. Kelly, Optimal Rough Terrain Trajectory Generation for Wheeled Mobile Robots, *Int. J. Robot. Res.* **26**(2), 141–166 (2007).
15. I. Yadav and H. G. Tanner, Reactive Receding Horizon Planning and Control for Quadrotors with Limited On-Board Sensing, in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems* (Las Vegas, NV, USA, 2020), pp. 7058–7063.
16. X. Liu, Z. Hu, Z. Yang and W. Zhang, Real-Time Control Allocation for Autonomous Surface Vehicle Using Constrained Quadratic Programming, *Guid. Navig. Control* **1**(04), 2140007 (2021).
17. Y. Okada and J. Miura, Exploration and Observation Planning for 3D Indoor Mapping, in *Proceedings of the IEEE/SICE International Symposium on System Integration* (Nagoya, Japan, 2015), pp. 599–604.
18. A. Bachrach, S. Prentice, R. He and N. Roy, RANGE–Robust Autonomous Navigation in GPS-Denied Environments, *J. Field Robot.* **28**(5), 644–666 (2011).
19. M. Ryll, J. Ware, J. Carter and N. Roy, Efficient Trajectory Planning for High Speed Flight in Unknown Environments, in *Proceedings of the International Conference on Robotics and Automation* (Montreal, QC, Canada, 2019), pp. 732–738.

Accepted Manuscript
Version of record at: https://doi.org/10.1142/S2737480722500200

20. M. Werling, J. Ziegler, S. Kammel and S. Thrun, Optimal Trajectory Generation for Dynamic Street Scenarios in a Frenét Frame, in *Proceedings of the IEEE International Conference on Robotics and Automation* (Anchorage, AK, USA, 2010), pp. 987–993.
21. I. Yadav and H. G. Tanner, Motion Planning and Visual-Inertial Target Tracking for UAV-Based Radiation Detection, in *Proceedings of the 28th Mediterranean Conference on Control and Automation* (Saint-Raphaël, France, 2020), pp. 1009–1014.
22. OpenCV Developers Team, Open Source Computer Vision (OpenCV) Library, http://opencv.org.
23. X. Zuo, Y. Yang, P. Geneva, J. Lv, Y. Liu, G. Huang and M. Pollefeys, LIC-Fusion 2.0: LiDAR-Inertial-Camera Odometry with Sliding-Window Plane-Feature Tracking, in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems* (Las Vegas, NV, USA, 2020), pp. 5112–5119.
24. W. Lee, K. Eckenhoff, Y. Yang, P. Geneva and G. Huang, Visual-Inertial-Wheel Odometry with Online Calibration, in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems* (Las Vegas, NV, USA, 2020), pp. 4559–4566.
25. J. Lv, J. Xu, K. Hu, Y. Liu and X. Zuo, Targetless Calibration of LiDAR-IMU System Based on Continuous-time Batch Estimation, in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems* (Las Vegas, NV, USA, 2020), pp. 9968–9975.
26. K. Eckenhoff, P. Geneva and G. Huang, MIMC-VINS: A Versatile and Resilient Multi-IMU Multi-Camera Visual-Inertial Navigation System, *IEEE Trans. Robot.* **37**(5), 1360–1380 (2021).
27. J. L. Owens, P. R. Osteen and K. Daniilidis, MSG-cal: Multi-Sensor Graph-Based Calibration, in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems* (Hamburg, Germany, 2015), pp. 3660–3667.