Highlights

Delayed Kalman Filter for Vision-Based Autonomous Flight in Ocean Environments

Kanishke Gamagedara, Taeyoung Lee, and Murray Snyder

- Design of vision-based multi-rotor unmanned aerial vehicle in ocean environment
- Sensor fusion of visual and inertial measurements using a delayed Kalman filter
- Autonomous flight experiments on a United States Naval Academy research vessel
- First demonstration for vision-based autonomous UAV landing on a ship in ocean

Delayed Kalman Filter for Vision-Based Autonomous Flight in Ocean Environments

Kanishke Gamagedara, Taeyoung Lee, and Murray Snyder

^a*Mechanical and Aerospace Engineering, The George Washington University, 800 22nd St NW, Washington, 20052, DC, USA*

Abstract

This paper presents the developments of flight hardware and software for a multirotor unmanned aerial vehicle, performing autonomous take-off and landing on a moving vessel in ocean environments. The proposed flight hardware is composed of a general-purpose computing module connected to a low-cost inertial measurement, an real-time kinematics GPS, motor speed controller, and a camera, through a custom-made printed circuit board. The flight software is developed in C++ with multi-threading such that the multiple tasks of control, estimation, and communication are executed simultaneously. The proposed flight system is verified with autonomous flight experiments on a United States Naval Academy research vessel operating in Chesapeake Bay. Two types of flight experiments are performed: autonomous flight utilizing real-time kinematics GPS for relative positioning, and vision-based autonomous flight, both for shipboard launch and landing.

Keywords: Multirotor unmanned aerial vehicle, Kalman filter, geometric control, ocean environment, autonomous flight

1. Introduction

The miniaturization of cost-effective and high-performing sensors and computing modules in recent years has widened the applications of unmanned aerial vehicle (UAV) systems. For example, various research groups have studied the UAV applications in agriculture [1, 2], aerial mapping and localization [3, 4], cooperative aerial transportation [5, 6, 7], gas leak identification [8], bridge inspection [9] and computer-vision based navigation [10, 11]. Although the most of such applications are focused on the above-land operations, deploying UAVs in ocean environments has numerous benefits.

Given the smaller size, maneuverability, lower cost, and longer range of UAVs, they can be extremely useful in scenarios such as search and rescue

---

missions, or reconnaissance operations in ocean. For example, a UAV system composed of multiple airborne and surface vessels to efficiently perform search and rescue operations is proposed in [12]. In [13], a UAV system attached to a ship is developed for automating search and rescue missions initiated by a detection of person overboard. Also, UAV systems have been used in harsh coastal areas for environment protection and surveillance operations, such as detecting sewage discharge and tracking trash [14, 15, 16]. Further, there has been a recent interest in measuring the ship air wake with anemometers mounted on a UAV [17, 18, 19] with applications mainly in the field of naval aviation. In addition, the benefits of using UAVs in various military and security applications in ocean environments such as aerial warfare, ordinance delivery, and patrolling have been studied in [20]

As illustrated by these applications, there are numerous application in operating a UAV in ocean environments with a ship as the base point. However, the UAV operation in ocean environment is challenging due to several reasons. First, the base point of the ship is moving along the water surface, while continuously rolling and pitching on the sea waves. This requires a more precise position control of the UAV when compared with above-ground operations. This is especially critical for the landing phase where the UAV is required to land on this rocking landing pad. Further, there can be unfavorable, strong wind and turbulences, specially in the air wake just behind a ship caused by the superstructure of the ship [21]. Also, vision-based sensors suffer from the high-dynamic range, caused by the fact that the ambient lighting varies significantly depending on the direction of the camera relative to the sun or the weather.

The main objective of this paper is to develop both software and hardware for an autonomous UAV platform that is capable of handling the harsh ocean environments, for safe landing on a desired target of a moving ship. The emphasis is on utilizing low-cost sensors with flight hardware and software platform that are designed and developed in house. As discussed above, an autonomous UAV landing on a moving ship requires an accurate estimation of the relative position of the UAV with respect to the ship. In the proposed UAV platform, two methods are utilized for relative positioning: an real-time kinematics (RTK) GPS-based solution and a vision-based solution. They have distinct characteristics in the accuracy and functionality. The RTK GPS achieves an centimeter-level accuracy, by compensating GPS signals from another receiver. But, this requires two GPS receivers, each mounted on the UAV and on the ship. In addition, a dedicated radio-based or network-based communication link between the two units should be established so that the ship-mounted unit can transmit its observations to the UAV-mounted unit. This separate communication link requires the UAV to carry additional power-consuming hardware, which in return reduces its payload. Further, the wireless communication between the ship and the UAV might cause security concern in the relevant mission scenarios as the radio signal may be corrupted by or expose its location to an adversary. The overall performance is significantly affected by the number and the configuration of GPS satellites that are available to provide reliable data.

To address these shortcomings and difficulties, the proposed UAV system was

extended to operate in a GPS-denied environment by a visual-inertial navigation technique. All image processing is performed in real-time, utilizing the onboard computer while running the estimation and control software in parallel. Since this does not require any communications from the ship, it does not suffer from the aforementioned drawbacks of using the RTK GPS-based positioning solution.

1.1. Prior Works

Here we summarizes the prior work related to autonomous UAV landing on a moving platform based on vision-based localization. Given that the GPS measurements typically have an error greater than one meter, the most of the prior works utilize vision-based approaches. One of vision based methods is visual servoing, where the distance from a known target is used to determine the position of the UAV. Visually guided landing has been studied in [22], where an image based solution has been used for horizontal positioning while using GPS altitude for vertical positioning. A UAV controller for autonomous landing on a moving platform using a monocular camera has been developed and tested with numerical simulations in [23]. Autonomous landing on a moving platform using image-based visual servoing has been studied in [24], which performs the most of the calculation on the 2D image space without the need for 3D position reconstruction. A similar method has been implemented in [25] to achieve quasi-stationary UAV position control. The authors of [26] performed a vision based landing on a moving truck in outdoor conditions, and the landing locations was denoted by a known marker with contrasting colors on the truck bed. Autonomous landing of a moving vehicle is studied in [27] using fiducial markers to identify the landing platform. Vision based landing on a moving platform has been tested in [28] using a single downward facing camera, where the landing pad emulates ship like motion which pitches and rolls with translations. This work assumes that the landing target is known, and the 3D positioning of the UAV with respect to the marker is determined. In [29], authors presented a vision-based landing method that would extend the range of the landing pad detection using cascaded fiducial markers. Autonomous takeoff and landing of UAV, both from a moving platform has been studied in [30]. The landing pad here is denoted by infra-red (IR) beacons, and the localization was performed with the onboard IR camera. Vision based landing pad detection with an indoor moving landing platform has been studied in [11] utilizing two cameras. Image-based visual servoing with a feed-forward approach has been used in [31] for vision based autonomous on landing platform attached to a truck. The landing platform is operated manually to simulate roll, pitch, and heave motion of a ship, and a fiducial marker has been used to identify the landing position. Similar visual servoing technique has been utilized in [32], where a UAV autonomous landing was tested on a landing deck mounted on a truck bed that can autonomously simulate ocean wave oscillations.

Another vision based method used for UAV localization is visual odometry. In previously discussed visual servoing methods, the UAV is required to have prior knowledge of the target to determine its positioning. In contrast, visual

odometry methods use features in the environments to determine the egomotion of the camera. However, these methods are computationally more expensive than visual servoing methods. An optical flow based visual odometry approach has been developed and used for autonomous UAV flight in [33]. Authors in [34] has proposed a method to combine both stereo-vision and optical flow to determine the visual odometry of a UAV. Given visual odometry techniques require sufficient illumination to work, a thermal camera based visual odometry approach has been used for a UAV flight in low visibility environments in [35]. Extended details of related vision-based methods for UAV navigation can be found in [36, 37].

Egomotion estimation may not be accurate if the motion is fast, and tricky lighting conditions such as low light or higher dynamic range may adversely affect the image-related measurements. In contrast, IMUs provide high frequency measurements that are more accurate in accelerations and angular velocities with higher magnitudes. Visual-inertial odometry (VIO) techniques leverage this complementary nature of the camera images and Inertial Measurement Unit (IMU) measurements to provide an improved positioning solution. VIO approaches can be divided into two main categories [38]: loosely-coupled approaches and tightly-coupled approaches.

In loosely-coupled approaches, a module processes image data to determine visual odometry, while a separate modules processes IMU measurements. Here, the image processing module determines position and the orientations, and the IMU module determines the position, velocity and the orientation. Each of those states are estimated in an independent manner, and fused together later to provide a more robust positioning solution. Since the each module is separate from each other, lightly-coupled techniques are easier to implement and less complicated. However, the accuracy of such methods are lower than the tightly-coupled approaches. Different loosely-coupled approaches have been proposed and implemented by various researchers [39, 40, 41].

In contrast, tightly-coupled approaches integrate IMU measurements directly with 2D image features. Such approaches are complex and difficult to implement, but provide high accurate positioning estimates. Tightly-coupled approaches may utilize either estimation methods or optimization methods to determine the solution. Estimations techniques such as [42, 43] use prediction and correction steps similar to an extended Kalman filter to provide a faster solution. Optimization based tightly-coupled VIO techniques [44, 38] estimate the position by minimizing the residual error of some objective function. Typical choice for this objective function would be the residual error of the equations of motion and the reprojection error. A model predictive control approach has been utilized in [45] to land a UAV on an unmanned surface vessel. The estimation structure includes a landing target predictor and a wave predictor to aid the UAV position during the landing phase.

As an alternative to strictly vision-based solutions, various researchers have experimented with ultra-wide band (UWB) based approaches for landing on moving platforms. Work presented in [46] uses vision-solution to detect the landing target, and an added UWB solution to improve it's accuracy. The

presented solution has been tested in software simulations. The authors in [47] use UWB positioning solution to land a UAV on a moving truck attached with a platform that can simulate sea conditions. Four UWB sensors were attached on the landing target, and a UWB sensor mounted on the UAV use them to localize itself.

1.2. Contributions

However, all of above discussed prior works in autonomous landing deal with above-ground scenarios or simulated ocean environments. There are limited results reporting autonomous flight in the actual ocean [48, 49]. The researchers in [48] has landed a hybrid vertical take-off and landing UAV autonomously on a moving ship. This work uses a combination of GPS and UWB to determine the position of the UAV relative to the ship, and the landing position error is in the range of $1-3\,\mathrm{m}$. Similarly, a tilt-rotor UAV has been been utilized in an autonomous shipboard landing on a moving ship [49]. This work uses an RTK GPS solution for more accurate positioning and has around $1\,\mathrm{m}$ mean landing position error. However, vision-based autonomous flight of a compact, multi-rotor UAV in the ocean environment has not been reported yet.

The main contribution of this paper is the developments of both low-cost hardware and software structure for a UAV system that can operate in an actual ocean environment, including autonomous landing on a ship. The proposed low-cost UAV platform is specifically designed to successfully operate with two relative localization schemes, namely RTK GPS and VIO, while compensating sensor delays and explicitly considering wind disturbances. We integrate a geometric adaptive controller for multi-rotor UAV that is robust against unknown disturbances with an extended Kalman filter that can handle time-delay in measurements, which is critical for RTK GPS relying on a communication link between two modules, and vision-based localization requiring non-trivial image processing.

Further, the proposed UAV system has been tested in autonomous flight experiments for launch and recovery in a United States Naval Academy research vessel. It should be emphasized that there is a great deal of unique challenges in performing autonomous flight experiments in ocean environments, such as strong air wakes and turbulence caused by the ship superstructures and adverse ambient lighting with high dynamic range, which cannot be accounted properly in simulated ocean environments. We completed successful flight experiments, where the mean position error is less than $15\,\mathrm{cm}$ during the entire landing phase, and the final horizontal landing accuracy is less than $12\,\mathrm{cm}$. In short, this paper reports the first vision-based autonomous shipboard launch and recovery for a multirotor UAV in ocean, thereby addressing the various issues of RTK GPS-based solutions.

2. Control and Estimation Schemes

To achieve safe autonomous flights in ocean environments, it is critical to utilize robust and reliable methods for control and estimation. For control, we
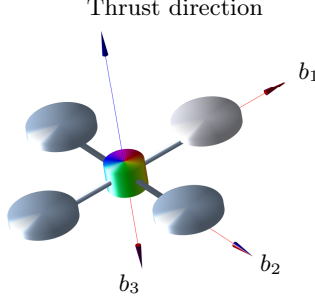
Figure 1: UAV model

implement a geometric adaptive controller with decoupled-yaw dynamics proposed in a prior work [50], which exhibits excellent performances under turbulent wind conditions by separating the roll/pitch dynamics critical for safety from the yaw dynamics.

In the proposed UAV system, all of the computation required for control and estimation is executed within the on-board computing module. Due to the limited computing power, there is non-negligible time delay in communicating with the base module in RTK GPS, or in processing the video images in vision-based localization. To compensate for this latency, the delayed Kalman filer developed for RTK GPS measurement [51] is adopted for sensor fusion, and it is further revised for vision-based localization.

2.1. Multirotor Dynamics

Consider the UAV shown in Figure 1. Let the mass of the UAV be $m \in \mathbb{R}$, and its moment of inertia with respect to the body frame be $J \in \mathbb{R}^{3 \times 3}$. The body-fixed frame $\mathcal{B} = \{b_1, b_2, b_3\}$ is defined such that its origin corresponds to the mass center of the UAV, and b_3 is pointing downward when hovering. The ship-fixed frame $\mathcal{I} = \{i_1, i_2, i_3\}$ is located at the center of the flight deck, and its first axis points toward the stern, and the third axis points downward the direction of gravity. Thus, the second axis is toward the port as illustrated in Figure 2. Note that the ship-fixed frame translates as the ship moves, but it does not roll or pitch: the first two axes are horizontal and the third axis is vertical always. Assuming that the ship does not accelerate or decelerate severely during the landing phase, it is assumed that the ship-fixed frame is inertial.

The position of the UAV is denoted by $x \in \mathbb{R}^3$, which is the vector from the origin of $\mathcal{I}$ to the origin of $\mathcal{B}$ resolved with respect to $\mathcal{I}$. Next, let $R \in \mathsf{SO}(3)$ be the attitude of the UAV. More precisely, R corresponds to the transformation of a representation of a vector from $\mathcal{B}$ to $\mathcal{I}$. The angular velocity of the UAV is defined as $\Omega \in \mathbb{R}^3$, and it is resolved with respect to $\mathcal{B}$.

The total thrust generated by all the rotors is denoted by $f \in \mathbb{R}$, and the resultant moment generated by the rotors, resolved in $\mathcal{B}$ is denoted by $M \in \mathbb{R}^3$.

6

Given the physical structure of a rotor-wing UAV, the total thrust f can only be generated along the direction of motor rotation axis of the UAV, namely $-b_3$ direction. This corresponds to $-Re_3$ in $\mathcal{I}$, where the i-th standard basis of $\mathbb{R}^3$ is denoted by $e_i \in \mathbb{R}^3$. Further, the gravity $g \in \mathbb{R}$ always acts on the UAV along the i_3 direction.

In practice, there are unmodeled dynamics or parametric errors, such as the error in the estimate of the mass (m) or inertia matrix (J). Further, there can be external disturbances such as wind acting upon the UAV. To account these, let $\mathbb{W}_x(x, v, R, \Omega)\theta_x \in \mathbb{R}^3$ be the modeling error and uncertainties in the translational dynamics, and let $\sum_{i=1}^3 \mathbb{W}_{R_i}(x, v, R, \Omega)\theta_{R_i}e_i \in \mathbb{R}^3$ be those of the rotational dynamics. Here, $\mathbb{W}_x(x, v, R, \Omega) \in \mathbb{R}^{3 \times P}, \mathbb{W}_{R_i}(x, v, R, \Omega) \in \mathbb{R}^{1 \times P}$ are any arbitrary function of the state that are assumed to be known, and $\theta_x, \theta_{R_i} \in \mathbb{R}^P$ are fixed unknown parameters, where P denotes the number of unknown parameters in θ_x and θ_{R_i}. The equations of motions can be written as

$$\dot{x} = v, \tag{1}$$

$$m\dot{v} = mge_3 - fRe_3 + \mathbb{W}_x(x, v, R, \Omega)\theta_x, \tag{2}$$

$$\dot{R} = R\hat{\Omega}, \tag{3}$$

$$J\dot{\Omega} + \Omega \times J\Omega = M + \sum_{i=1}^3 \mathbb{W}_{R_i}(x, v, R, \Omega)\theta_{R_i}e_i, \tag{4}$$

where the *hat* map, $\wedge : \mathbb{R}^3 \to \mathfrak{so}(3)$ is defined such that $\hat{x}y = x \times y$ and $\hat{x}^T = -\hat{x}$ for any $x, y \in \mathbb{R}^3$. This is represented throughout the paper either with $\hat{\cdot}$ or $(\cdot)^\wedge$.
 In the above equations, (1)-(2) correspond to the translational dynamics of the UAV, while (3)-(4) correspond to the rotational dynamics. It should be noted that above equations of motion hold for any conventional multi-rotor UAV such as a quadrotor, a hexrotor or an octocopter, as long as the thrust of each rotor can be converted into the total thrust f and the moment M in the body-fixed frame.

2.2. Geometric Adaptive Control

Suppose that the desired position trajectory is given by $x_d(t) \in \mathbb{R}^3$, and also the desired direction of the first body fixed axis is prescribed as $b_{1_d}(t) \in \mathsf{S}^2 = \{r \in \mathbb{R}^3 \,|\, \|r\| = 1\}$. Then, the control objective is to determine the thrust of each rotor such that the actual trajectory $x(t)$ follows its desired value $x_d(t)$ asymptotically. To achieve this control objective, an adaptive geometric controller with decoupled-yaw dynamics is utilized [50].

In the multirotor dynamics summarized by (1)–(4), the translational dynamics (1) and (2) are coupled with the attitude dynamics (3) and (4) only through the single term, namely $-Re_3$ in (2), which corresponds to the direction of the total thrust determined by the third body-fixed axis b_3. As such, the rotation about b_3, referred to as the yaw dynamics does not affect the translational dynamics directly.
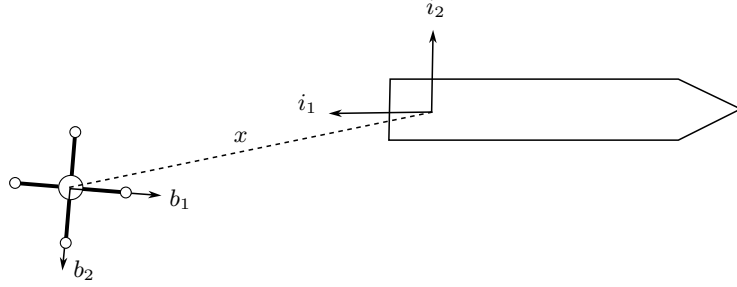
Figure 2: Frame definitions

The potential issue of the yaw dynamics is that it is actuated by the weak reactive torque from each rotor. More specifically, to generate the reactive torque about b_3 for the yaw motion, two motors rotating in the same direction as the required yaw rotation need to be decelerated rapidly, while other two motors need to be accelerated. As the mass inertia of the propeller is small, it is often required to rotate them excessively even for a moderate yaw moment. This may amplify the undesirable effects caused by input saturation, motor/propeller imbalances, or errors in motor calibrations. More importantly, such issues may deteriorate the roll/pitch dynamics that are critical for the stability of position dynamics of the UAV.

To avoid these problems, a position control system for a UAV that decouples the yaw control from the roll/pitch control for position tracking is utilized. A roll/pitch controller is designed such that the direction of the thrust follows its desired value. Then, an yaw controller is constructed separately to control the rotation about the thrust direction. These are globally formulated on the two-dimensional unit sphere and on the one-dimensional circle, respectively, to avoid singularities and complexities inherent to local coordinates. Further, both position and attitude controllers include adaptive terms to compensate for unmodeled dynamics and external disturbances.

To achieve these, it is assumed that the mass distribution of the UAV is symmetric about b_3 such that the inertia matrix is given by $J = \text{diag}[J_1, J_1, J_3]$ for $J_1, J_3 > 0$. This is required to separate the yaw dynamics, and it is satisfied in general as the UAVs such as quadrotors and octocopters are rotationally symmetric. Under this assumption, the roll/pitch dynamics and be decoupled from the yaw dynamics. Specifically, let $\Omega = (\Omega_1, \Omega_2, \Omega_3) \in \mathbb{R}^3$ and $M = (M_1, M_2, M_3) \in \mathbb{R}^3$. Then, it can be shown that (4) can be re-written into a set

of three scalar equations:

$$J_1\dot{\Omega}_1 = -(J_3 - J_1)\Omega_2\Omega_3 + M_1 + \mathbb{W}_{R_1}\theta_{R_1}, \tag{5}$$

$$J_1\dot{\Omega}_2 = (J_3 - J_1)\Omega_3\Omega_1 + M_2 + \mathbb{W}_{R_2}\theta_{R_2}, \tag{6}$$

$$J_3\dot{\Omega}_3 = M_3 + \mathbb{W}_{R_3}\theta_{R_3}. \tag{7}$$

Here (5)-(6) correspond to the roll/pitch dynamics, and (7) corresponds to the yaw dynamics decoupled from the other parts.

Let the direction of the resultant thrust along the negative third body-fixed axis $b_3 = Re_3 \in \mathsf{S}^2$, and the desired direction of thrust $b_{3_d}(t) \in \mathsf{S}^2$, The roll/pitch error variables can be defined as $e_b = b_{3_d} \times b_3$ and $e_\omega = \omega_{12} + \hat{b}_3^2\omega_{12_d}$, with $\omega_{12} = \Omega_1 b_1 + \Omega_2 b_2 \in \mathbb{R}^3$. Then, a fictitious control $\tau \in \mathbb{R}^3$ can be defined as

$$\tau = (-J_3\Omega_2\Omega_3 + M_1)b_1 + (J_3\Omega_3\Omega_1 + M_2)b_2. \tag{8}$$

Further, let $\bar{\theta}_{R_i} \in \mathbb{R}^P$ for $i \in \{1,2\}$ be the estimated parameter, which is updated according to

$$\dot{\bar{\theta}}_{Ri} = \gamma_R \mathbb{W}_{R_i}^T(e_\omega + c_2 e_b) \cdot b_i, \tag{9}$$

for $c_2, \gamma_R \in \mathbb{R} > 0$.

Let the position error be $e_x = x - x_d \in \mathbb{R}^3$ and let the velocity error $e_v = v - \dot{x}_d \in \mathbb{R}^3$. Also, let $\bar{\theta}_x \in \mathbb{R}^P$ be an estimate to the unknown parameter θ_x, which is updated according to the following adaptive law for the translational dynamics:

$$\dot{\bar{\theta}}_x = \begin{cases} \gamma_x \mathbb{W}_x^T(e_v + c_1 e_x) & \text{if } (\|\bar{\theta}_x\| < B_{\theta_x}), \text{ or} \\ & (\|\bar{\theta}_x\| = B_{\theta_x} \text{ and } \bar{\theta}_x^T\mathbb{W}_x^T(e_v + c_1 e_x) \le 0), \\ \gamma_x(I - \frac{\bar{\theta}_x\bar{\theta}_x^T}{\bar{\theta}_x^T\bar{\theta}_x})\mathbb{W}_x^T(e_v + c_1 e_x) & \text{otherwise,} \end{cases} \tag{10}$$

for $c_1, \gamma_x \in \mathbb{R} > 0$. Further, the corresponding estimation error $\tilde{\theta}_x \in \mathbb{R}^P$ is defined as $\tilde{\theta}_x = (\theta_x - \bar{\theta}_x)$.

Then, define following control laws:

$$f = -(-k_x e_x - k_v e_v - \mathbb{W}_x\bar{\theta}_x - mge_3 + m\ddot{x}_d) \cdot b_3, \tag{11}$$

$$\begin{aligned}\tau = &-k_b e_b - k_\omega e_\omega - J_1(b_3 \cdot \omega_{12_d})\dot{b}_3 - J_1\hat{b}_3^2\dot{\omega}_{12_d} \\ &- \mathbb{W}_{R_1}\bar{\theta}_{R_1}b_1 - \mathbb{W}_{R_2}\bar{\theta}_{R_2}b_2,\end{aligned} \tag{12}$$

for controller gains $k_x, k_v, k_b, k_\omega > 0$. Considering the UAV dynamics given by (2) and (4), choosing the total thrust f to be (11), and the moments (M_1, M_2) to be defined by (8) and (12), it can be shown that the equilibrium of the tracking errors for the roll/pitch dynamics, namely $(e_x, e_v, \tilde{\theta}_x, e_b, e_w, \tilde{\theta}_{R_1}, \tilde{\theta}_{R_2}) = (0,0,0,0,0,0,0)$ is stable in the sense of Lyapunov, and the tracking errors e_x, e_v, e_b, e_w asymptotically converge to zero [50].

For the yaw dynamics, the error terms are defined as $e_y = -b_{1_c} \cdot b_2$, and $e_{\omega_y} = \Omega_3 - \omega_{c_3}$. Here, $b_{1_c} \in \mathsf{S}^2$ is the projection of the desired direction of b_1, $b_{1_d} \in \mathbb{R}^3$, to the plane normal to b_3, and is defined as $b_{1_c} = (I_{3\times 3} - b_3 b_3^T) b_{1_d}$, and $\omega_{c_3} = b_3 \cdot (b_{1_c} \times \dot{b}_{1_c}) \in \mathbb{R}$. Considering positive controller gains k_y, k_{ω_y}, the control moment is chosen as

$$M_3 = -k_y e_y - k_{\omega_y} e_{\omega_y} + J_3 \dot{\omega}_{c_3} - \mathbb{W}_{R_3} \bar{\theta}_{R_3}, \tag{13}$$

where $\bar{\theta}_{R_3} \in \mathbb{R}^P$ is an estimate to θ_{R_3} and it is updated according to

$$\dot{\bar{\theta}}_{R_3} = \gamma_R \mathbb{W}_{R_3}{}^T (e_{\omega_y} + c_3 e_y), \tag{14}$$

for $c_3 \in \mathbb{R} > 0$. Then, for positive controller gains k_y, k_{ω_y}, it can be shown that the equilibrium $(e_y, e_{\omega_y}, \tilde{\theta}_{R_3}) = (0, 0, 0)$ is stable in the sense of Lyapunov, and $e_y, e_{\omega_y} \to 0$ as $t \to 0$. Again, the proofs are relegated to [50] for brevity.

The roll/pitch controller defined in (11)-(12) and the yaw controller defined in (13) were implemented and utilized for the trajectory control of the UAV during the autonomous operations in ocean environments.

2.3. Delayed Kalman Filter

For the vision-based autonomous UAV landing, it is required to estimate the UAV position with respect to the ship with a monocular camera. The objective is to adopt a method that is robust and computationally-efficient for real-time localization with the onboard computing. As such, an optimization-based tightly-coupled visual-inertial odometry method was utilized for vision processing [38] to integrate inertial measurements and image data, while using semi-direct visual odometry (SVO) proposed in [52] as the image processing front end.

Specifically, the RPG-SVO open-source library [53] has been adopted. The approach here combines both feature-based indirect methods and sparse direct methods, and then utilizes the IMU measurements to estimate the visual odometry. This enables vision-based pose estimation without any specific predefined fiducial markers. This library was modified to be compiled and run optimally at 5 Hz in the Jetson TX2, which is based on the ARM architecture. While the localization in the RPG-SVO library was fast compared to other existing visual-inertial odometry methods, there exists a non-trivial delay (about 500 ms) as it is executed on the low powered Jetson TX2 running multiple parallel threads.

In contrast, the measurements from the IMU are not delayed significantly and they are available at a higher rate of 200 Hz. Directly fusing the delayed vision measurements with non-delayed IMU measurements is non-optimal, and can cause the estimation to diverge over time. As such, the delayed Kalman filter developed in [51] for fusing delayed RTK GPS measurements was adopted for processing vision measurements in this work.

Consider the UAV dynamics derived in Section 2.1. Assume that the IMU is mounted at the center of gravity of the UAV, and the rotation matrix from the IMU frame to the $\mathcal{B}$-frame is denoted by $R_{bi} \in \mathsf{SO}(3)$. Let the acceleration

10

measurement from the IMU be a_{IMU}, which is corrupted by an additive noise $w_a \in \mathbb{R}^3$, and a bias $b_a \in \mathbb{R}$ along the direction of gravity. Further, let the angular velocity measurements from the IMU be $\Omega_{\text{IMU}} \in \mathbb{R}^3$, with an additive noise $w_\Omega \in \mathbb{R}^3$. Both of these measurements are treated as an exogenous time-varying signals resolved in the $\mathcal{B}$-frame. Then, the equations of motion for (x, v, R) are given by

$$\dot{x} = v, \tag{15}$$

$$\dot{v} = RR_{bi}(a_{\text{IMU}} + w_a) + e_3 b_a + g e_3, \tag{16}$$

$$\dot{R} = R\left(R_{bi}(\Omega_{\text{IMU}} + w_\Omega)\right)^\wedge. \tag{17}$$

2.3.1. Prediction

Suppose the distribution at $t = t_k$, given by $(\bar{x}_k, \bar{v}_k, \bar{R}_k, \bar{b}_{a_k})$ and their co-variance matrix, P_k. The mean values $(\bar{x}, \bar{v}, \bar{R})$ are propagated by discretizing (15)–(17) in the absence of the process noise via the following second order explicit method:

$$\bar{x}_{k+1} = \bar{x}_k + h_k \bar{v}_k + \frac{h_k^2}{2} \bar{a}_k, \tag{18}$$

$$\bar{v}_{k+1} = \bar{v}_k + \frac{h_k}{2}(\bar{a}_k + \bar{a}_{k+1}), \tag{19}$$

$$\bar{R}_{k+1} = \bar{R}_k \exp\left\{\frac{h_k}{2}(\bar{\Omega}_k + \bar{\Omega}_{k+1})^\wedge\right\}, \tag{20}$$

$$\bar{b}_{a_{k+1}} = \bar{b}_{a_k}, \tag{21}$$

where $h_k = t_{k+1} - t_k$ is the discrete time step, and

$$\bar{\Omega}_k = R_{bi}\Omega_{\text{IMU}_k}, \tag{22}$$

$$\bar{a}_k = \bar{R}_k R_{bi} a_{\text{IMU}_k} + e_3 \bar{b}_{a_k} + g e_3. \tag{23}$$

Consider small perturbations to $(\bar{x}, \bar{v}, \bar{b}_a) \in \mathbb{R}^{3+3+1}$, which are denoted by $(\delta x, \delta v, \delta b_a) \in \mathbb{R}^{3+3+1}$, respectively. For the attitude, $R = \bar{R}\exp(\hat{\eta})$, for $\eta \in \mathbb{R}^3$, which is uniquely determined when the angle between R and $\bar{R}$ is less than π radians. This yields the following perturbation of the attitude

$$\delta R = R\hat{\eta}. \tag{24}$$

Let $\mathbf{x} \in \mathbb{R}^{10}$ be defined as

$$\mathbf{x} = (\delta x, \delta v, \eta, \delta b_a). \tag{25}$$

Further, the covariance of $\mathbf{x}$, $P \in \mathbb{R}^{10 \times 10}$, is given by

$$P = \text{E}[\mathbf{x}\mathbf{x}^T].$$

In short, the uncertainty distribution of the state is defined by the mean $(\bar{x}, \bar{v}, \bar{R}, \bar{b}_a)$, and the covariance P.

Considering above discretized equations of motion in (15)–(17), it can be shown that the perturbed system can be rearranged into a matrix form as

$$d\mathbf{x} = (A(t)\mathbf{x} + F_1(t)\mathbf{w}_1)dt + F_2 dW_{b_a}, \tag{26}$$

where $\mathbf{w}_1 = [w_a; w_\Omega] \in \mathbb{R}^6$, and the matrices $A(t) \in \mathbb{R}^{10 \times 10}$, $F_1(t) \in \mathbb{R}^{10 \times 6}$, $F_2 \in \mathbb{R}^{10 \times 1}$ are given by

$$A(t) = \begin{bmatrix} 0_{3\times3} & I_{3\times3} & 0_{3\times3} & 0_{3\times1} \\ 0_{3\times3} & 0_{3\times3} & -R(R_{bi}a_{\mathrm{IMU}})^\wedge & e_3 \\ 0_{3\times3} & 0_{3\times3} & -(R_{bi}\Omega_{\mathrm{IMU}})^\wedge & 0_{3\times1} \\ 0_{1\times3} & 0_{1\times3} & 0_{1\times3} & 0 \end{bmatrix},$$

$$F_1(t) = \begin{bmatrix} 0_{3\times3} & 0_{3\times3} \\ RR_{bi} & 0_{3\times3} \\ 0_{3\times3} & R_{bi} \\ 0_{1\times3} & 0_{1\times3} \end{bmatrix},$$

$$F_2 = \begin{bmatrix} 0_{9\times1} \\ B_a \end{bmatrix}.$$

In (26), $w_{b_k} \in \mathbb{R}$ follows the distribution defined by $w_{b_k} \sim \mathcal{N}(0, h_k)$. Defining $\mathbf{w}_k = [w_{a_k}; w_{\Omega_k}; w_{b_k}] \in \mathbb{R}^7$, it can be shown that $\mathbf{w}_k \sim \mathcal{N}(0, Q_k)$, and its covariance $Q_k \in \mathbb{R}^{7 \times 7}$ is

$$Q_k = \mathrm{diag}[Q_a, Q_\Omega, h_k]. \tag{27}$$

Following [54, p 330], the discrete-time linearized equation can be written as

$$\mathbf{x}_{k+1} = A_k\mathbf{x}_k + F_k\mathbf{w}_k, \tag{28}$$

where $A_k \in \mathbb{R}^{10 \times 10}$ and $F_k \in \mathbb{R}^{10 \times 7}$ are

$$A_k = I_{10\times10} + h_k A(t_k)\Psi, \tag{29}$$
$$F_k = h_k \Psi[F_1(t_k), F_2], \tag{30}$$

with a matrix $\Psi \in \mathbb{R}^{10 \times 10}$

$$\Psi = I_{10\times10} + \frac{h_k}{2}A(t_k)\times$$
$$\left(I + \frac{h_k}{3}A(t_k)\left(I + \cdots \left(I + \frac{h_k}{N}A(t_k) \right) \right) \right),$$

for an integer $N \geq 1$. Finally, from (28), the covariance is propagated as

$$P_{k+1} = A_k P_k A_k^T + F_k Q_k F_k^T. \tag{31}$$

When the prediction is executed, the mean values of the states are propagated with (18)–(21), and the covariance matrix is propagated with (31). Note that the prediction step requires the IMU measurements $(\Omega_{\mathrm{IMU}}, a_{\mathrm{IMU}})$ at (22) and (23).

12

2.3.2. Non-Delayed IMU Correction

Next, the state correction is performed using IMU attitude measurements that has no delay. The expected attitude measurement is given by

$$\bar{R}_{\mathrm{IMU}} = \bar{R} R_{bi}. \tag{32}$$

The residual error, or the discrepancy between (32), includes two error sources: the attitude estimation error represented as $R = \bar{R} \exp(\hat{\eta})$ for $\eta \in \mathbb{R}^3$, and the sensor measurement error, $\zeta_R \in \mathbb{R}^3$. This can be mathematically represented as

$$R_{\mathrm{IMU}} = \bar{R} \exp \hat{\eta} R_{bi} \exp \hat{\zeta}_R. \tag{33}$$

Given that the attitude evolves on $\mathsf{SO}(3)$, the residual error, $\delta z \in \mathbb{R}^3$, is represented at the tangent space of $\mathsf{SO}(3)$ at $\bar{R}$ as

$$R_{\mathrm{IMU}} = \bar{R} \exp(\delta \hat{z}) R_{bi}, \tag{34}$$

Equation (34) can be re-arranged to show that

$$\delta z = \frac{1}{2} (\bar{R}^T R_{\mathrm{IMU}} R_{bi}^T - R_{bi} R_{\mathrm{IMU}}^T \bar{R})^{\vee}. \tag{35}$$

Using $\exp(\widehat{R\zeta}) = R \exp \hat{\zeta} R^T$ for any $R \in \mathsf{SO}(3)$ and $\zeta \in \mathbb{R}^3$, (33) is rewritten as

$$R_{\mathrm{IMU}} = \bar{R} \exp \hat{\eta} \exp(\widehat{R_{bi}\zeta_R}) R_{bi}, \tag{36}$$

which is compared with (34) to obtain

$$\exp(\delta \hat{z}) = \exp \hat{\eta} \exp(\widehat{R_{bi}\zeta_R}). \tag{37}$$

According to the BCH formula [55], and after ignoring higher-order terms, it can be shown that

$$\delta z = \eta + R_{bi}\zeta_R.$$

This can be re-arranged to

$$\delta z = H\mathbf{x} + G\mathbf{v}, \tag{38}$$

where $\mathbf{v} = \zeta_R \in \mathbb{R}^{3 \times 3}$, and $H \in \mathbb{R}^{3 \times 10}$ and $G \in \mathbb{R}^{3 \times 3}$ are defined as

$$H = [0_{3 \times 3}, \, 0_{3 \times 3}, \, I_{3 \times 3}, \, 0_{3 \times 1}], \quad G = R_{bi}. \tag{39}$$

Now, the posterior mean and the posterior covariance are given by

$$\bar{\mathbf{x}}^+ = K\delta z, \tag{40}$$
$$P^+ = (I_{10 \times 10} - KH)P. \tag{41}$$

13

where $K \in \mathbb{R}^{10 \times 3}$ and $S \in \mathbb{R}^{3 \times 3}$ are

$$K = PH^T S^{-1}, \quad S = HPH^T + GV_R G^T.$$

Then, the posterior mean of each state is updated as

$$\bar{x}^+ = \bar{x} + \delta\bar{x}^+, \quad \bar{v}^+ = \bar{v} + \delta\bar{v}^+, \tag{42}$$

$$\bar{R}^+ = \bar{R}\exp(\hat{\bar{\eta}}^+), \quad \bar{b}_a^+ = \bar{b}_a + \delta\bar{b}_a^+. \tag{43}$$

In summary, when a new IMU measurement is available, the residual δz is computed by (35), and the mean and the covariance are updated following (40)–(43).

2.3.3. Delayed Vision Position Correction

Given the processing latency of the onboard computing module, the vision-based localization measurements are considered to have a known time delay t_D. Let the delay time period be $t_{k-D} = t_k - t_D$, for $D \in \mathbb{Z}^+$, and position and velocity measurement error covariance values be $V_x, V_v \in \mathbb{R}^{3 \times 3}$. Then, the vision measurement equations can be written as

$$z_{k-D} = H\mathbf{x}_{k-D} + \zeta, \tag{44}$$

where $z = [x; v]^T \in \mathbb{R}^6$, and $\xi = [\zeta_x; \zeta_v]^T \in \mathbb{R}^6$ with $\xi \sim \mathcal{N}(0_{6\times1}, V)$ for $V = \mathrm{diag}(V_x, V_v) \in \mathbb{R}^{6\times6}$. The matrix $H \in \mathbb{R}^{6 \times 10}$ is

$$H = \begin{bmatrix} I_{3\times3} & 0_{3\times3} & 0_{3\times4} \\ 0_{3\times3} & I_{3\times3} & 0_{3\times4} \end{bmatrix}. \tag{45}$$

The objective of the vision correction step is to determine the current state, (x_k, v_k, R_k, b_{a_k}), conditioned by the vision measurement in the past, z_{k-D}. This can be optimally calculated by correcting the prior state at t_{k-D} using the new vision measurement z_{k-D}, and propagating it to t_k by repeating the IMU attitude correction step and the prediction step. Let the prior distribution given all the measurements up to t_{k-D} be defined by the mean $(\bar{x}_{k-D}, \bar{v}_{k-D}, \bar{R}_{k-D}, \bar{b}_{a_{k-D}})$ and the covariance P_{k-D}. Let the posterior distribution conditioned by z_{k-D} denoted by the superscript $+$. From (44),

$$\bar{\mathbf{x}}_{k-D}^+ = K(z_{k-D} - [\bar{x}_{k-D}; \bar{v}_{k-D}]), \tag{46}$$

$$P_{k-D}^+ = (I_{10\times10} - KH)P_{k-D}. \tag{47}$$

where $K \in \mathbb{R}^{10 \times 6}$ is given by

$$K = P_{k-D}H^T S^{-1}, \quad S = HP_{k-D}H^T + V.$$

When a new vision measurement arrives, the saved prior state at t_{k-D} is corrected with (46)–(47), and the posterior state is propagated up to the current time step, t_k, by repeating the prediction and the IMU correction using the saved IMU measurements during the delay period, t_D.

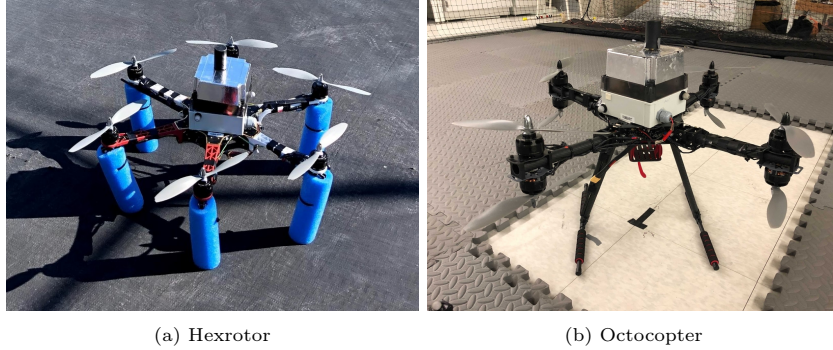(a) Hexrotor   (b) Octocopter

Figure 3: UAVs designed and developed for the hardware flight tests

The above approach preserves the optimality in the estimation. However, given the requirements for saving prior states and the non-delayed measurements during the delay period, there is an increased memory requirement. This can grow large as the delay amount is increased relative to the non-delayed measurement frequency. For example, in the developed UAV system, the IMU measurements were processed at 200 Hz, while vision measurements were acquired at 5 Hz with a delay of 500 ms. This requires saving and processing 100 measurements for the delayed vision correction. This burden can be minimized by sub-sampling IMU measurements over the prediction. The effects of the sub-sampling on the trade-off between the computational load and the accuracy for delayed GPS measurements were discussed and in [51]. In that work, sub-sampling reduced the delayed GPS correction (5 Hz measurements with 400 ms latency) time to around 70% of non-sub-sampled case, while only increasing the average position error by around 1%. Similar computational gains were noticed with 5 Hz vision measurements with 500 ms latency.

3. Flight Hardware Developments

This section details the development of a UAV hardware platform that is capable of flying in ocean environments. The proposed platform was designed such that it could be easily transformed into different configurations such as quadrotors or hexrotors with minimal changes, and it is composed of both off-the-shelf and custom designed components. A hexrotor and an octocopter used for the results included in the paper are shown in Figure 3. Unless noted otherwise, the details in this section are valid for any UAV platform used, regardless of the motor/frame configuration.

3.1. Sensors and Actuators

A 9-axis VectorNav VN100 [56] sensor was used as the onboard inertial measurement unit (IMU), which provides the UAV acceleration, the angular velocity, and the attitude. The sensor communication was achieved through a

3.3 V serial communication line (UART port), and it was able to provide all required measurements at 200 Hz without any noticeable latency.

A relatively low-cost SwiftNav Piksi Multi RTK GPS receiver [57] was used to determine the position and the velocity of the UAV. In contrast to the conventional single point precision GPS receivers, RTK GPS provides centimeter-level accuracy. This higher accuracy is achieved by using another RTK GPS unit, namely the base receiver, sending corrections to the UAV RTK GPS receiver, namely the rover receiver. Further, given that the onboard GPS receiver receives the base GPS position, they can provide accurate relative position measurements. This is particularly useful when the base is moving and the relative position of the UAV is more important rather than the absolute position, such as operating the UAV off of a moving ship. Similar to the IMU, the GPS receiver communicates over a 3.3 V serial communication (UART port) line, and provides measurements at 5 Hz. However, the correction calculation required to achieve the centimeter-level accuracy is computationally expensive and the processing time causes about a delay of 0.4 s to the GPS measurements.

As the base receiver should send corrections to the rover receiver, it is required to have a dedicated communication link between them. A set of 915 MHz FreeWave telemetry [58] units were paired with each of the RTK GPS receivers to establish this radio communication. Each GPS receiver has a separate serial port to communicate with their respective telemetry unit. The base receiver sends its observations to the base telemetry, which transmits the observations to the rover telemetry as a radio signal. Upon receiving the transmitted message, the rover telemetry communicates the observations to the rover receiver through its serial port.

For the vision-based localization of the UAV, a Point Grey Chameleon [59] camera was mounted on the UAV. This is a monocular grayscale camera with a global shutter. Although the camera was capable of handling a USB 3.0 connection, a USB 2.0 cable was used between the camera and the computing module to avoid interference with the GPS signals. Further, the camera frame rate was reduced to 15 Hz given the limited bandwidth of the USB 2.0 connection.

The only actuators onboard the UAV are the motors. For the developed UAV systems, 700 KV T-Motor [60] DC brush-less electrical motors were used. They were combined with 10×4.7 APC propellers. Each brush-less DC motor is connected to an electronic speed controller (ESC), which modulates the motor power to realize the commands sent from the computing module. In the hexrotor platform, MikroKopter BL-Ctrl [61] v2 modules were used as ESCs, while MikroKopter BL-Ctrl v3 modules were used on the octocopter. Both ESCs support I2C communication, though which the throttle commands are sent from the computing module to the ESCs. The main difference between the two ESC types is that the v3 modules are capable of motor braking as opposed to the v2 module.

3.2. Computing Module and the PCB

The UAV platform was developed with two main components: rover and base. The term *rover* is used to denote all components onboard the UAV in-

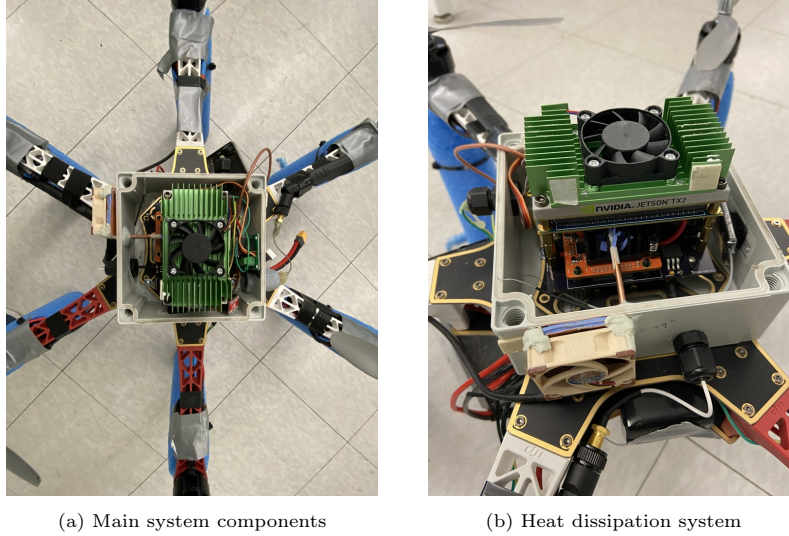(a) Main system components (b) Heat dissipation system

Figure 4: Internal components of the system, mounted on a hexrotor

cluding the sensors, actuators, and the onboard computer, whereas the term *base* refers to the components on the base point of operations, including the base computer, base GPS receiver and base IMU on the ship.

An NVIDIA Jetson TX2 module running Ubuntu 18.04 with JetPack 4.4 was used as the onboard computer. The multiple tasks of reading from sensors, computing all the calculations required for the UAV estimation and control, and controlling the actuators were performed on this computing module. The onboard computer was fitted to a ConnectTech Orbitty carrier board and all peripherals were connected with the TX2 through it. This board was selected based on the smaller footprint and the input/output port availability. The TX2 with a green-colored heat sink attached can be observed in Figure 4, and the carrier board can be seen right below the TX2. To connect the computing module to the sensors and actuators described in Section 3.1, a custom printed circuit board (PCB) was developed. This PCB provided mounting points for all the sensors, and voltage regulation required for electronic components.

3.3. Other Hardware Components

A communication link between the UAV and the base computer is required in order to send commands and monitor flight data in real-time. This is achieved through a Wi-Fi connection, where the UAV is configured to act as a Wi-Fi hotspot to which the base computer is connected. However, if the UAV is too far from the base, this connection may become unreliable. This can cause the connection to be dropped, which is not desirable for the robust operation of the UAV. In order to address this, a Redport Halo Wi-Fi extender was utilized. The extender is configured to connect to the UAV hotspot, and the base

computer is connected to the extender through an ethernet cable. This significantly improved the signal strength and the reliability of the UAV-base Wi-Fi connection.

The above-mentioned communication link is used to send critical commands such as motor on/off signals or commanded flight mode of takeoff or hover. As such, if the Wi-Fi connection is lost in any reason, there is no way to control the UAV. For an additional safety measure, a radio controlled relay switch is installed at the I2C connection from the carrier board to the ESCs, which is activated by a conventional UAV remote control (RC) radio to turn off the motors independently from other parts of the system. Further, the relay is configured such that the motors are turned off automatically if the power is lost or the UAV flies beyond the range of the RC.

The UAV platform was designed with the goal of the operating it in ocean environments. As such, it was expected that there is a risk of unexpected UAV water landings. To recover the UAV in such cases, the landing legs of the UAV were extended and the pool noodles were wrapped around the extension, as shown in blue color of Figure 3a. This modification results in the 2.5-to-1 ratio between the buoyancy and the weight so that the UAV can float. Further, the critical components of the rover are enclosed in a water tight enclosure (see the silver-white box in Figure 3). However, this raised another issue that the system overheats during flight operations, especially over the summer. All of the RTK GPS, the computing module, and the power regulators generate heat when they operate. Given that the enclosure is water-tight, the heat does not dissipate effectively through the box in a passive manner. While an exhaust fan would be helpful, it will destroy the desired impenetrability of water as well. Nothing that the RTK GPS unit is the main source of heat, an active heat rejection system was designed. A copper heat pipe was connected to the heatsink on the GPS receiver, and it was routed to the outside of the box, while keeping the box watertight with a thermal-resistant compound. A fan was connected to the pipe through a copper plate to actively remove heat from it. This heat dissipation system can be seen in Figure 4b. Further, the silver colored reflective layer was added to the top of the enclosure to mitigate the radiation from the sun.

3.4. Combined Hardware Platform

All the components discussed so far together constitute the proposed UAV hardware platform. However, when the UAV is being operated on a ship, it is also estimate its position and attitude relative to the ship. As such, a separate UAV sensor platform with the same PCB was mounted on the ship and it is connected to the base laptop computer. The base sensor data were saved directly on the computer, together with the time-stamped data received from the UAV through the Wi-Fi connection.

The overall system configuration is illustrated in Figure 5, which presents the interconnection between components (solid:wired, dotted:wireless) and the power (red). The components inside the light blue area are mounted on the PCB, and for the UAV, all the components except the ESCs and the motors are
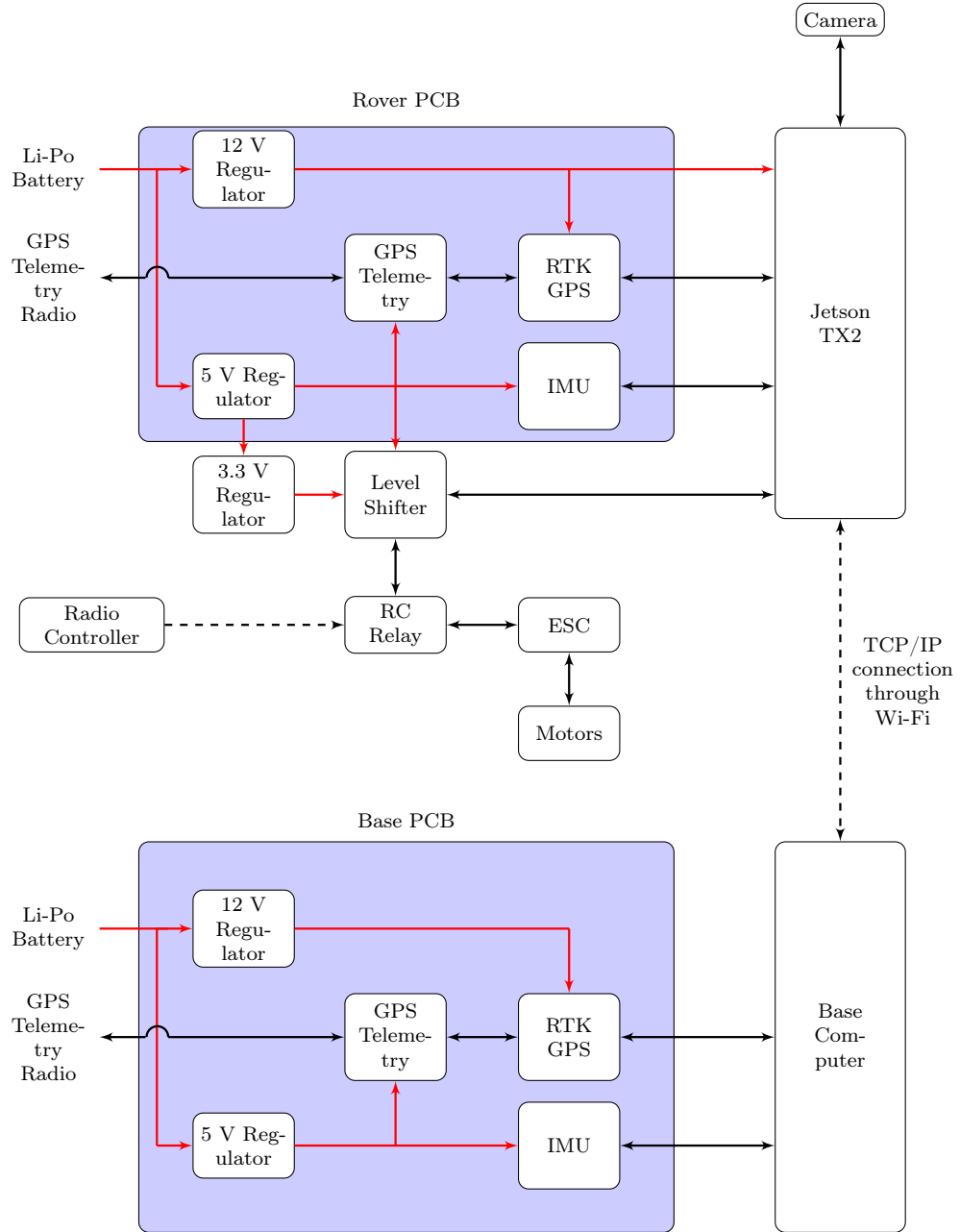
Figure 5: Schematic of the rover-base system: solid black - wired communication, dashed black - wireless communication, solid red - power

placed inside the water-tight enclosure. The TCP/IP communication connection between base computer and the rover is routed via the Wi-Fi extender as detailed in the previous section. The data link between the base PCB and the base computer were established through two FTDI cables: one for the base IMU and another for the base RTK GPS receiver.

This combined hardware platform has been utilized in multiple UAVs with varying motor configurations. The particular choice of the motor configuration was determined based on the operating environment of the UAV and the payload. A hexrotor frame was used for RTK GPS-based flight tests, where the UAV was carrying the water-tight enclosure, the heat dissipation system, and the pool noodles for flotation. During the vision-based flight tests, the added weight of the camera made the hexrotor UAV to operate closer to the motor saturation limits, especially inside the turbulent ship air wake. To overcome this, the frame was upgraded to an octocopter setup with co-axial rotors. The upgraded octocopter configuration provided a greater thrust-to-weight ratio for safe and stable UAV flight experiments in ocean with the additional vision hardware components.

4. Flight Software Structures

The hardware platform designed in Section 3 operates with a flight software running in its computing module. It was developed with a C++ code utilizing multi-threaded programming, which implements two main components: a geometric adaptive controller described in Section 2.2 to control the motion of the multirotor UAV, and the extended Kalman filter (EKF) detailed in Section 2.2 to estimate the state of the UAV, while compensating sensor delay.

As described in Section 3.4, the developed UAV platform can be divided into two modules of the rover and the base. The code was split into two parts as well where each module worked in tandem to achieve the control objective. However, it should be emphasized that all UAV calculations were performed inside the computer onboard the UAV, and the base computer only handled sending commands (e.g.: motor on/off, trajectory mode changes) and receiving data for real-time monitoring and logging. The function of each module is detailed in the following sections.

4.1. Rover Code

The rover code utilized multiple threads using the C++ standard threading library to execute various tasks simultaneously. Each onboard sensor is dedicated to a separate thread to communicate with the Jetson TX2 computing module, namely IMU thread, GPS thread, and camera thread. All of the sensor interfaces are asynchronous, meaning each sensor pushes the sensor message whenever a new measurement is available, without need to constantly check the time for synchronization. When a sensor thread receives a new asynchronous measurement, it is saved into a first-in-first-out (FIFO) buffer with a time stamp, which is referred to as sensor message buffer. This buffer is shared with all sensor threads, and as such, its data structure was designed to ensure thread-safety.
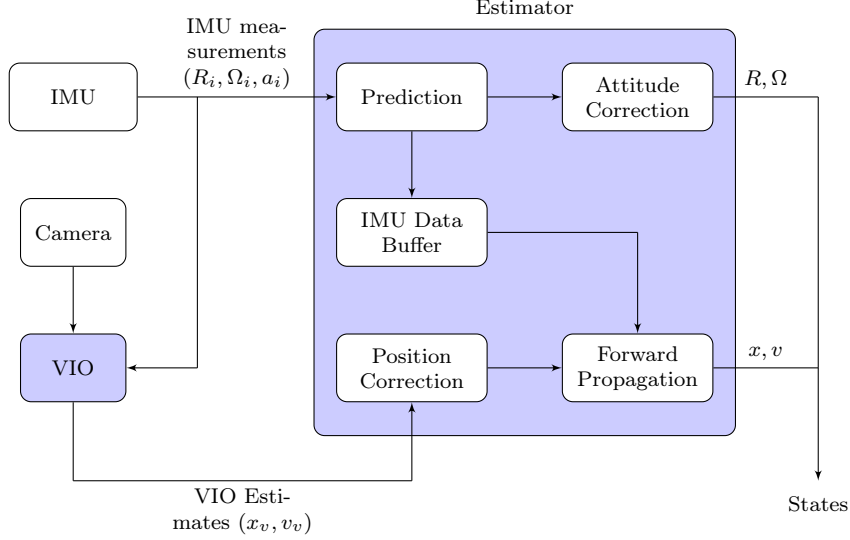
Figure 6: Integration of vision measurement into the delayed Kalman filter

One of the critical component of the rover code was the implementation of the vision processing. The RPG-SVO library used for vision processing requires image stream and the IMU measurements for its calculations. The IMU thread reads and processes the measurements from the IMU, and the camera thread was used to read image frames from the camera. The IMU thread was augmented such that whenever IMU sends a new sensor message, it immediately publishes the message in a format that is compatible with the RPG-SVO library. A similar publisher was added to the camera thread where it publishes the images read from the camera in a format understandable to the RPG-SVO library.

When the RPG-SVO library gets new sensor messages, it processes them to determine the camera position with respect to the initial frame and publishes it. On the onboard computer, this process takes around $500\,\mathrm{ms}$ when all the other threads from the UAV code are being executed. A separate thread was created to subscribe to this message, which treated it as a delayed measurement when integrating it to the estimator, as described in Section 2.3. Specifically, sub-sampled IMU measurements were saved during the delay period, and once the processed VIO position message arrived, it was used to correct the saved prior state. Then, the saved IMU measurements were used to forward propagate the corrected state to the current time horizon, compensating for the VIO processing delay. This process is illustrated in Figure 6.

The controller was executed in a separate thread, which takes the states updated by the estimator and the desired commands defined by user inputs. The controller generates the desired motor command values, and pushes them into another thread-safe FIFO buffer, namely motor command buffer. The motor control thread reads from the motor control buffer and executes necessary steps

to rotate the motors at the required speed.

In addition to the above described threads, another thread, namely the Wi-Fi thread runs on the rover. The purpose of this thread is to receive commands such as motor on/off, trajectory mode changes (takeoff, land, etc.) from the base computer through a Wi-Fi connection. In addition, the estimated UAV states on the rover are sent to the base for real-time data monitoring and logging.

The FIFO buffer method described above minimizes the scope of the required threadlocks, allowing multiple threads to access the shared data promptly and safely. The corresponding average frequencies of the critical control thread and the motor threads are around $400\,\mathrm{Hz}$ and $800\,\mathrm{Hz}$, respectively.

4.2. Base Code

The base code is similar with the above rover code, except that it does not utilize any thread for estimation, control, and motor. But, it still uses sensor threads for data monitoring. Also, the base code runs its own Wi-Fi thread that is responsible for sending commands to the rover, and receiving internal state data from the rover for real-time visualizing and logging.

Further, the base code includes two other threads: the graphical user interface (GUI) thread and the data logging thread. As the name suggests, the GUI thread displays a graphical user interface that can be used to visualize the UAV states such as position, attitude, and motor commands, and take commands from the user, which is developed with the gtkmm library [62]. The data logging thread logs data from both of the rover and the base for post flight analysis.

5. Flight Experiments in Ocean Environments

With the UAV hardware platform described in Section 3 running the software described in Section 4, we performed several autonomous flight experiments in ocean environments. These are executed on research vessels of the United States Naval Academy, specifically YP689 or YP700 depending on availability (see Figure 7), over Chesapeake Bay, Maryland. These results are presented in two sub-sections: autonomous UAV operation with RTK GPS-based positioning, and with vision-based positioning, respectively.

5.1. Flight Experiments with RTK GPS-Based Positioning

During the underway flight tests, the base module was mounted on the ship and its sensors were connected to the base computer through USB extension cables to read and save the ship's status. The UAV was configured to be a Wi-Fi hotspot and the base computer was connected to it for sending commands and for monitoring real-time UAV data. The RTK GPS sensor measurements were configured such that the initial position of the UAV on the ship corresponds to the origin of the ship-fixed frame for relative localization. The first axis of the ship-fixed frame points toward the stern or the aft-most part of the ship,

and the second axis points the port side. According to the right handed rule, the third axis points downward.

The flight experiment was performed at the flight deck of the aft. During the experiment, the ship was moving at around 5 knots, mostly along a straight line such that the headwind makes a 0° angle with the ship's direction of motion. The motor arming and the UAV take-off was performed manually. After the take-off, the desired position is specified by manual key strokes on the base laptop. Six keys on the keyboard are assigned to the axes of the ship-fixed frame and the opposite directions, and each stroke results in the command of shift along the corresponding axis by 10 cm. Using this, the UAV was commanded to fly about 5 m behind the stern. After maintaining the relative position for 4 s, the command for landing was transmitted, after which the desired landing trajectory was calculated and followed.

The autonomous landing mode generates the desired trajectory such that the UAV moves toward the origin at a constant velocity, while keeping an offset in the vertical direction. Once the UAV has reached this waypoint, it performs a quick descent to reach the origin of the trajectory, thereby completing the autonomous landing. More specifically, let $x_{l_0} \in \mathbb{R}^3$ be the UAV relative position at $t = 0$ when the landing command was issued, and let $x_{dsc} \in \mathbb{R}^3$ be the point about the flight deck where the descending for landing starts. Both x_{l_0} and x_{dsc} are defined in the ship-fixed frame, $\mathcal{I}$, as detailed in Section 2.1. The desired landing trajectory is selected to connect the three points of x_{l_0}, x_{dsc}, and $x = 0$ with a fixed velocity. Specifically, the desired landing trajectory $x_l(t)$ is given by

$$x_l(t) = \begin{cases} x_{l_0} + v_{wp}t & \text{if } t \leq \frac{\|x_{dsc} - x_{l_0}\|}{\|v_{wp}\|} \\ x_{dsc} & \text{else if } \|x_{dsc} - x\| > \epsilon_x \\ x_{dsc} + v_{dsc}t & \text{otherwise,} \end{cases} \tag{48}$$

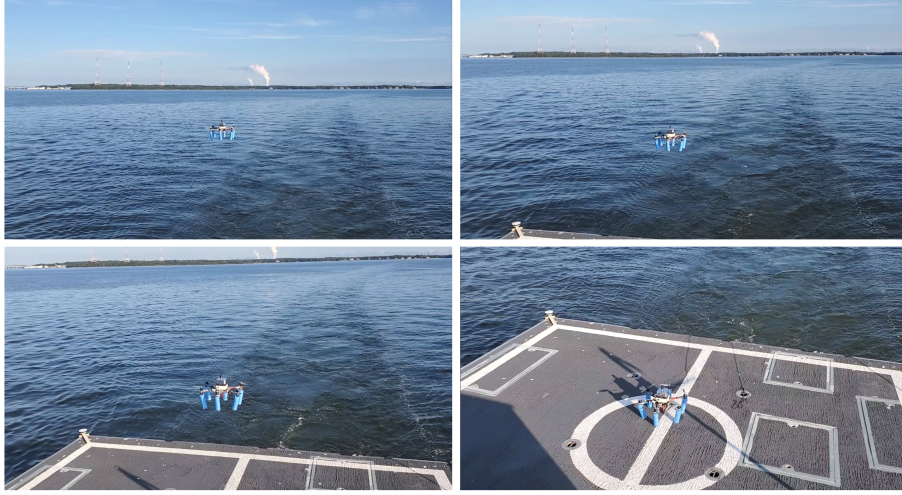where $v_{wp} \in \mathbb{R}^3$ is defined such that it is parallel to $x_{dsc} - x_{l_0}$ with the pre-

23

Figure 8: Screenshots of the video during the autonomous shipboard landing

scribed magnitude of $\|v_{wp}\| = 0.4\,\mathrm{m\,s^{-1}}$. After the UAV arrives at x_{dsc} with the tolerance of $\epsilon_x = 0.1\,\mathrm{m}$, it descends with the velocity of $v_{dsc} = [0, 0, 0.2]^T \mathrm{m\,s^{-1}}$.

The control parameters are chosen as

$$m = 2.95,\ J = \mathrm{diag}(0.03, 0.03, 0.04),$$
$$k_x = \mathrm{diag}(10, 10, 15),\ k_v = \mathrm{diag}(12, 12, 12),$$
$$k_b = 3,\ k_\omega = 0.8,\ k_y = 0.8,\ k_{\omega_y} = 0.4$$

Further, both $\mathbb{W}_x$ and $\mathbb{W}_R$ were set to identity.

Figure 8 presents a few screenshots from the video taken at an experiment, and the full length video is available at [63]. During the experiment, there was a cross-wind of approximately 6 knots, and consequently, the UAV tilts toward the crosswind to follow the command while overcoming the effects of wind.

While the video shows that the UAV is connected to a tether, it should be noted that this was not a tethered flight. The sole purpose of the tether was to make the recovery of the UAV easier in case of a water landing.

The results for this flight test is presented in Figure 9, where the black lines show the measurements, and the red lines show the estimated states. The blue lines represent the desired value of the states, if applicable. The overall 3-dimensional trajectory is shown in Figure 9a, and the relative position is presented in Figure 9c. As described above, the UAV was moved to the aft of the ship manually, and then switched to the autonomous landing trajectory, which is defined by (48), at around 65 s. As a result, the desired position before the switching is composed of a series of step functions. It can be seen that the hexrotor UAV manages to successfully follow the desired trajectory and land at the origin.
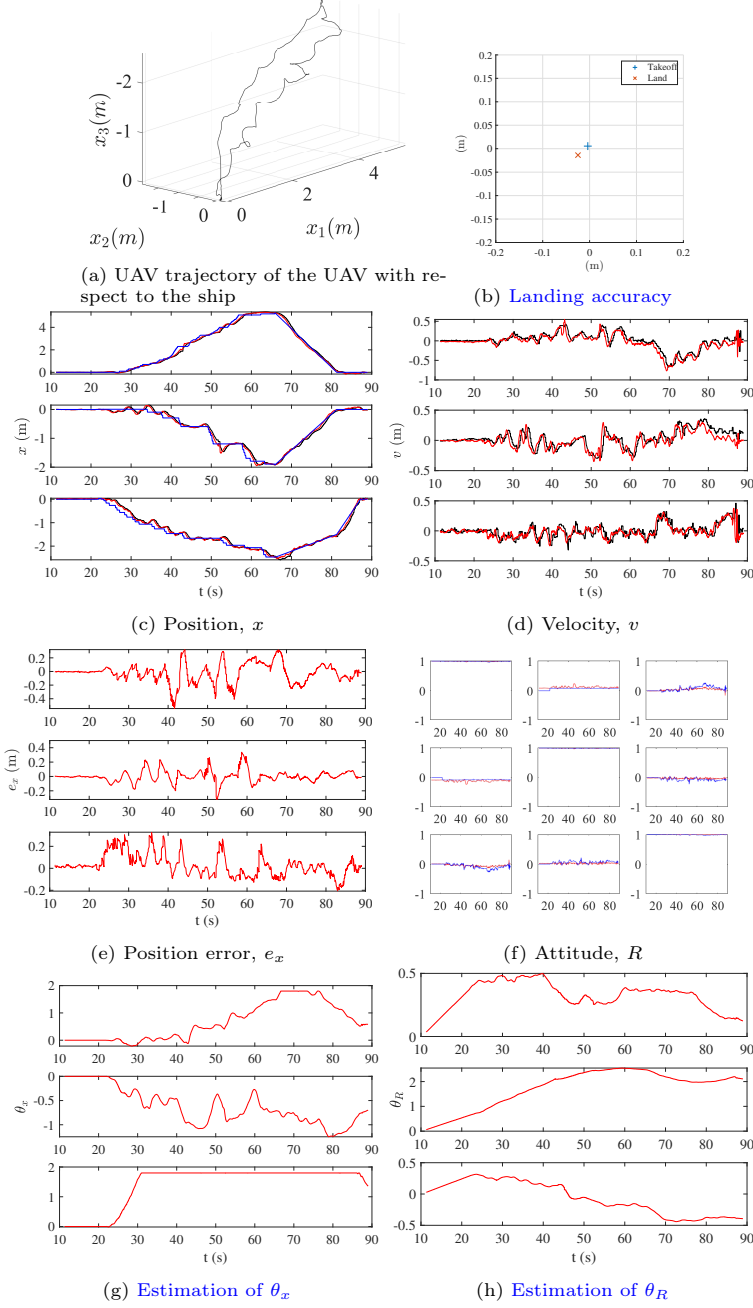
(a) UAV trajectory of the UAV with respect to the ship

(b) Landing accuracy

(c) Position, x

(d) Velocity, v

(e) Position error, e_x

(f) Attitude, R

(g) Estimation of θ_x

(h) Estimation of θ_R

Figure 9: Results for autonomous flights onboard the ship with RTK-GPS based positioning: red - estimated, black - measurements, blue - desired

(a) Octocopter on the ship deck
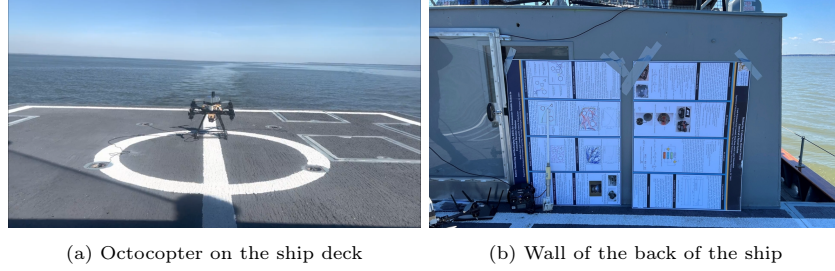
(b) Wall of the back of the ship

Figure 10: Octocopter UAV onboard the ship

The measured velocity and the estimated one of the UAV are presented in Figure 9d. The time shift between the measured position (black line) and the estimated position (red line) is due to the delay correction in the estimator. This time shift is also present in the position plot, but it is difficult to observe given the scale of the axes.

The error between the actual position and the desired position, namely $e_x = x - x_d$ is presented in Figure 9e. For the total duration of the flight, the mean position error was $E[\|e_x\|] = 16$ cm. During the autonomous landing phase after $65\,$s where the trajectory was smooth, it was about $14\,$cm. The final horizontal landing error, that is the magnitude of the difference between the landing target and the actual landing point, is about $3\,$cm.

Next, the nine elements of the rotation matrix representing the attitude of the UAV are presented in Figure 9f. The each subplot here represents the value of each element of the 3×3 rotation matrix, as a function of time. It can be observed that the UAV closely follows the desired attitude computed by the controller to follow the desired position trajectory. During the flight test, the mean attitude error was $3.72°$. Here, the attitude error vector is defined by $e_R = 0.5(R_d^T R - R^T R_d)^\vee$, and the desired attitude, R_d, was provided by the position controller.

Further, the estimation of unknown parameter for both position (θ_x) and attitude (θ_R) are presented in Figure 9g and Figure 9h, respectively. 5

5.2. Flight Experiments with Vision-Based Positioning

Next, with the VIO solution of the RPG-SVO library integrated to the UAV flight software as described in the Section 4.1, the UAV was tested in real-time flight tests onboard a moving ship. The sensors and other hardware configuration was identical to Section 5.1, except for the octocopter UAV frame and the added camera. The camera was mounted under the frame such that it was facing toward the ship during the flight. The UAV with the mounted camera resting on the ship deck is shown in Figure 10a.

For octocopter control parameters, following values were used:

$$m = 3.4, \ J = \mathrm{diag}(0.036, 0.036, 0.039),$$
$$k_x = \mathrm{diag}(12, 12, 13), \ k_v = \mathrm{diag}(13, 13, 18),$$
$$k_b = 2.5, \ k_\omega = 0.8, \ k_y = 0.8, \ k_{\omega_y} = 0.4$$

Further, both $\mathbb{W}_x$ and $\mathbb{W}_R$ were set to identity.

During the takeoff and the landing phases of the flight, the UAV camera was too close to the vertical wall of the ship's superstructure that has almost no texture. Consequently, when the camera is pointed to this feature-less wall for a long time period, the VIO solution degrades and diverges. To avoid that, two posters with random images and texts were attached to the walls as shown in Figure 10b. This improved the VIO, particularly for take-off and landing, when the UAV was close to this wall.

With the above setup, the octocopter UAV was tested in multiple flight tests onboard the US Naval Academy research vessel YP700 at Chesapeake Bay. Similar with the previous flight experiments, the UAV was moved to the aft of the ship manually, until the flight mode is switched to landing. The processed camera images, together with the pose estimated by the VIO, were sent to the base computer and saved for visualization in the post processing. This saved video and a separate video recorded from the ship were stitched together after the flight tests, and it is shared at [64]. Two videos were synced with the fixed frame rates, and the time sync may not be exact. A set of screenshots of the same video is shown in Figure 11.

The states of the UAV during the flight tests are presented in Figure 12. In these plots, black lines represent measurements, red lines represent the estimated states and the blue lines represent the desired value of the states where applicable.

Further, it should be emphasized that all the computations, including UAV control and image processing with VIO, were performed on the onboard computing module in real-time.

Figure 12c shows the desired trajectory with the position determined by the VIO, and the position estimated by the delayed Kalman filter compensating the time delay of the onboard VIO. Each axis of this plot represents the position of the UAV in the local north-east-down frame, where the origin is located at the center of the ship's landing deck. The octocopter UAV was moved to the aft of the ship by changing the position set points through operator commands until 114 s and the step-like increases in the desired commands in the plots represent these manual command changes. Then, the trajectory mode was switched to the autonomous land mode at around 115 s, and the mode change can be noticed by the smooth desired trajectory lines in this plot. This trajectory here is as same as the trajectory described in (48), but with $x_{dsc} = [0, 0, -0.7]^T$ m. Further, the shift in the estimated position and the VIO position along the time axis is a result of the time delay present in the VIO measurements. As it can be observed, despite the time delay, the delayed Kalman filter estimates the correct position, and the UAV closely follows the desired trajectory.

27

Figure 11: Screenshots from the flight test video - flow is left-to-right and top-to-bottom
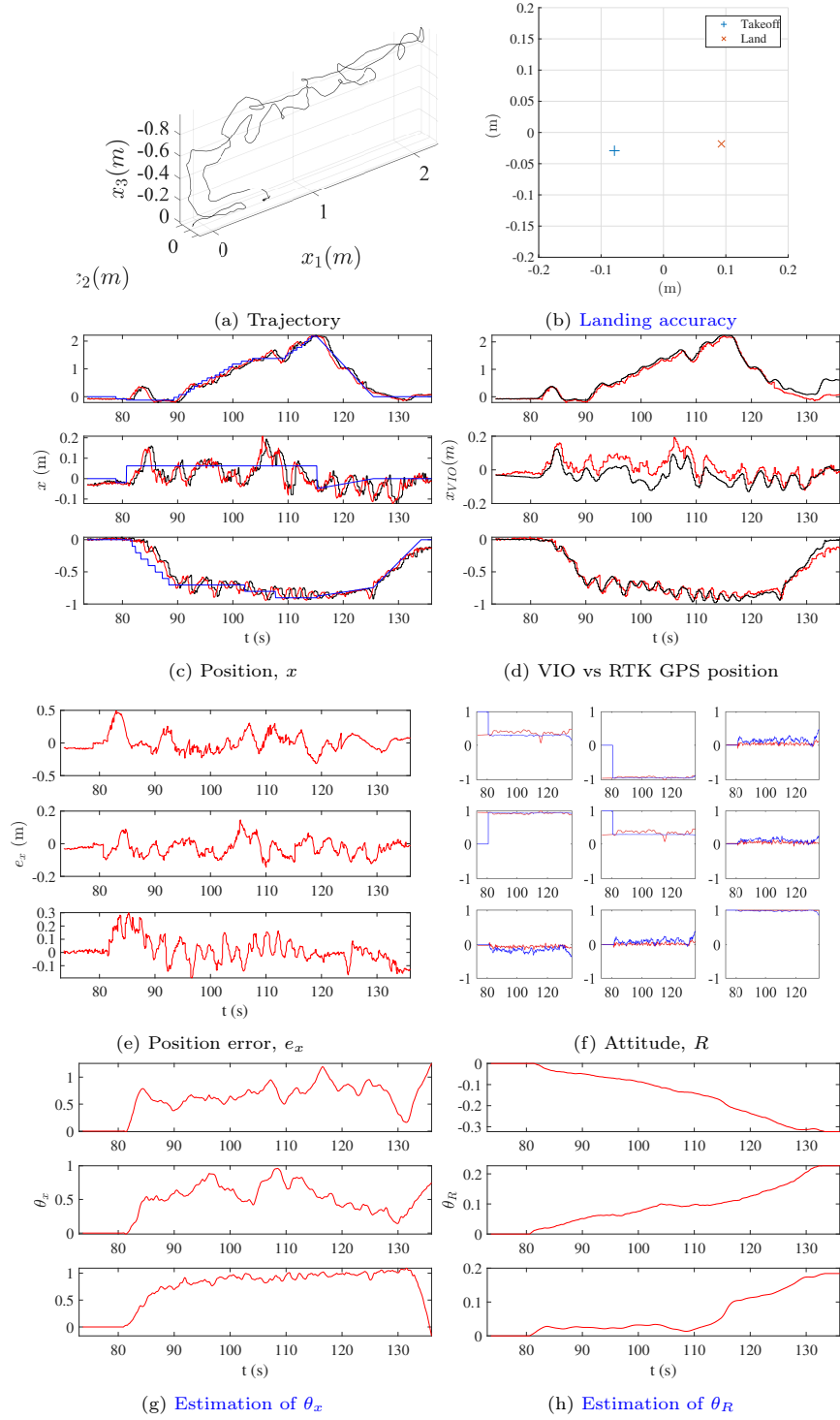
(a) Trajectory

(b) Landing accuracy

(c) Position, x

(d) VIO vs RTK GPS position

(e) Position error, e_x

(f) Attitude, R

(g) Estimation of θ_x

(h) Estimation of θ_R

Figure 12: Results for autonomous flights with vision-based positioning: red - estimated, black - measurements, blue - desired

29

The position estimated by the VIO and the position measured by the RTK GPS unit are presented in Figure 12d. It should be noted that due to the processing delay, the RTK GPS measurements are delayed by $400\,\mathrm{ms}$ while the VIO estimations are delayed by $500\,\mathrm{ms}$. There is a minor offset along the time axis between them due to that, but the estimated VIO position reasonably follows the measured RTK position that is considered as a ground truth in accessing the accuracy of the VIO. It should be noted that the RTK GPS measurements were collected only for the comparison, and they were not used in the state estimation.

The position error, $e_x = x - x_d$, is shown in Figure 12e. For the total duration of the flight, the mean position error was about $15\,\mathrm{cm}$. During the autonomous landing phase after $115\,\mathrm{s}$ where the trajectory was smooth, the mean position error was about $12\,\mathrm{cm}$. Both of these error magnitudes are at the same level with the results presented in Section 5.1 based on the RTK GPS. Further, the final horizontal landing error, that is the magnitude of the difference between the landing target and the actual landing point, is about $17\,\mathrm{cm}$.

The attitude plots are presented in Figure 12f, where each column in the subplot represents the corresponding column in the rotation matrix. In this plot, desired attitude shown as a blue line is determined by the position controller. It can be observed that the actual attitude closely follows the desired attitude, making the accurate position control viable. During the flight test, the mean attitude error was $5.22°$. Here, the attitude error vector is defined by $e_R = 0.5(R_d^T R - R^T R_d)^\vee$, and the desired attitude, R_d, was provided by the position controller.

Further, the estimation of the unmodeled dynamics of both position dynamics and attitude dynamics are shown in Figure 12g and Figure 12h, respectively.

In short, these report the first vision-based autonomous launch and recovery of a multirotor UAV in ocean environments.

5.3. Maritime Flight Experiments

From the presented flight experiments, we observed several situations that are unique to ocean environments, as listed below. These might be a consideration for any future research in relevant topics.

- Occasionally, the yaw angle measured by the onboard IMU was noticeably off, which caused the relative position estimate unreliable. This might have been caused by the large steel structures or the communication equipment of the vessel. To address this, we monitored the IMU measurement closely before each flight experiment, and we re-calibrated the IMU as needed.

- In ocean environments, there are great variations in light levels, depending on the orientation of the UAV relative to the Sun, the elevation angle of the Sun, and the weather conditions. For example, when the camera is pointed directly toward the Sun, the presented VIO fails. The robustness of VIO against high dynamic range and weather is potentially improved by adopting additional sensors, such as radar [65].

- The ocean wakes and the cloud may generate spurious visual features that may degrade the reliability of VIO. Further, the research vessel is mostly featureless by design. As such, any VIO based on visual features may be complemented by techniques based on the overall shape of the vessel, which is the current research efforts of the authors.

6. Conclusions

This paper presents the development of a UAV platform that can be operated in ocean environments. Both of flight hardware and software are developed in house with specific consideration to handle autonomous flight in ocean. The flight hardware is based on a general-purpose computing module connected to IMU, RTK GPS, camera, and motor speed controller over a custom designed PCB. For flight in ocean environments, the enclosure is designed to be watertight, while actively dissipating heat. The flight software is developed by multi-thread programming in C++, implementing an extended Kalman filter that compensates measurement delays and a geometric adaptive control to handle wind disturbances. The proposed UAV platform has been successfully tested in above-sea flight experiments in mild winds and clear/overcast conditions on a US Naval Academy research vessel, with two cases of RTK GPS positioning and vision-based approach. Despite the myriad of prior work related to autonomous landing of UAVs, there has not been any reported autonomous launch and recovery with vision-based approaches in over-ocean environments. The average position accuracy during the vision-based landing was about 12 cm, and the final landing error was less than 10 cm Further, the final positioning accuracy achieved by the proposed method during the RTK GPS-based landing is in sub 15 cm range, compared with higher than 1 m positioning errors in other GPS-based existing works on autonomous UAV shipboard landing in ocean environments. The future works include improving the robustness of the vision-based localization in the ocean when the number of salient features is limited.

Funding Sources

References

[1] J. Kim, S. Kim, C. Ju, H. I. Son, Unmanned aerial vehicles in agriculture: A review of perspective of platform, control, and applications, IEEE Access 7 (2019) 105100–105115. doi:10.1109/access.2019.2932119.

[2] D. C. Tsouros, S. Bibi, P. G. Sarigiannidis, A review on UAV-based applications for precision agriculture, Information 10 (11) (2019) 349. doi:10.3390/info10110349.

[3] J.-C. Trujillo, R. Munguia, S. Urzua, E. Guerra, A. Grau, Monocular visual SLAM based on a cooperative UAV–target system, Sensors 20 (12) (2020) 3531. doi:10.3390/s20123531.

[4] E. Kaufman, K. Takami, Z. Ai, T. Lee, Autonomous quadrotor 3D mapping and exploration using exact occupancy probabilities, in: 2018 Second IEEE International Conference on Robotic Computing (IRC), IEEE, 2018. doi:10.1109/irc.2018.00016.

[5] D. S. D'antonio, G. A. Cardona, D. Saldaña, The catenary robot: Design and control of a cable propelled by two quadrotors (Feb. 2021). arXiv:2102.12519.

[6] B. Yu, K. Gamagedara, S. Kim, T. Lee, J. Suk, Geometric control and experimental validation for a quadrotor UAV transporting a payload, in: 2020 59th IEEE Conference on Decision and Control (CDC), IEEE, 2020. doi:10.1109/cdc42340.2020.9303889.

[7] R. Cotsakis, D. St-Onge, G. Beltrame, Decentralized collaborative transport of fabrics using micro-UAVs, in: 2019 International Conference on Robotics and Automation (ICRA), IEEE, 2019. doi:10.1109/icra.2019.8793778.

[8] T. Barchyn, C. H. Hugenholtz, S. Myshak, J. Bauer, A UAV-based system for detecting natural gas leaks, Journal of Unmanned Vehicle Systems (oct 2017). doi:10.1139/juvs-2017-0018.

[9] S. Chen, D. F. Laefer, E. Mangina, S. M. I. Zolanvari, J. Byrne, UAV bridge inspection through evaluated 3d reconstructions, Journal of Bridge Engineering 24 (4) (apr 2019). doi:10.1061/(asce)be.1943-5592.0001343.

[10] W. Youn, H. Ko, H. Choi, I. Choi, J.-H. Baek, H. Myung, Collision-free autonomous navigation of a small UAV using low-cost sensors in GPS-denied environments, International Journal of Control, Automation and Systems (Aug. 2020). doi:10.1007/s12555-019-0797-7.

[11] D. Falanga, A. Zanchettin, A. Simovic, J. Delmerico, D. Scaramuzza, Vision-based autonomous quadrotor landing on a moving platform, in: 2017 IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR), IEEE, 2017. doi:10.1109/ssrr.2017.8088164.

[12] T. Yang, Z. Jiang, R. Sun, N. Cheng, H. Feng, Maritime search and rescue based on group mobile computing for unmanned aerial vehicles and unmanned surface vehicles, IEEE Transactions on Industrial Informatics 16 (12) (2020) 7700–7708. doi:10.1109/tii.2020.2974047.

[13] V. A. Feraru, R. E. Andersen, E. Boukas, Towards an autonomous UAV-based system to assist search and rescue operations in man overboard incidents, in: 2020 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), IEEE, 2020. doi:10.1109/ssrr50563.2020.9292632.

[14] S. O'Young, P. Hubbard, RAVEN: A maritime surveillance project using small UAV, in: 2007 IEEE Conference on Emerging Technologies & Factory Automation (EFTA 2007), IEEE, 2007. doi:10.1109/efta.2007.4416878.

[15] C. Chen, F. Lyu, Unmanned-system-based solution for coastal submerged outfall detection, in: 2021 7th International Conference on Hydraulic and Civil Engineering & Smart Water Conservancy and Intelligent Disaster Reduction Forum (ICHCE &SWIDR), IEEE, 2021. doi:10.1109/ichceswidr54323.2021.9656282.

[16] Y.-H. Liao, J.-G. Juang, Real-time UAV trash monitoring system, Applied Sciences 12 (4) (2022) 1838. doi:10.3390/app12041838.

[17] G. Loubimov, M. P. Kinzel, S. Bhattacharya, Measuring atmospheric boundary layer profiles using UAV control data, in: AIAA Scitech 2020 Forum, American Institute of Aeronautics and Astronautics, 2020. doi:10.2514/6.2020-1978.

[18] K. Gamagedara, T. Lee, M. R. Snyder, Real-time kinematics GPS based telemetry system for airborne measurements of ship air wake, in: AIAA Scitech 2019 Forum, American Institute of Aeronautics and Astronautics, 2019. doi:10.2514/6.2019-2377.

[19] A. Kumar, P. Ben-Tzvi, Novel wireless sensing platform for experimental mapping and validation of ship air wake, Mechatronics 52 (2018) 58–69. doi:10.1016/j.mechatronics.2018.04.009.

[20] G. C. Rosa, M. M. Marques, V. Lobo, Unmanned Aerial Vechicles in the Navy: Its Benefits, Scientific Bulletin of Naval Academy 19 (1) (2016) 39–43. doi:10.21279/1454-864x-16-i1-007.

[21] C. J. Mallon, B. J. Muthig, K. Gamagedara, K. Patil, C. Friedman, T. Lee, M. R. Snyder, Measurements of ship air wake using airborne anemometers, in: 55th AIAA Aerospace Sciences Meeting, American Institute of Aeronautics and Astronautics, 2017. doi:10.2514/6.2017-0252.

[22] S. Saripalli, J. Montgomery, G. Sukhatme, Visually guided landing of an unmanned aerial vehicle, IEEE Transactions on Robotics and Automation 19 (3) (2003) 371–380. doi:10.1109/tra.2003.810239.

[23] J. Ghommam, M. Saad, Autonomous landing of a quadrotor on a moving platform, IEEE Transactions on Aerospace and Electronic Systems 53 (3) (2017) 1504–1519. doi:10.1109/taes.2017.2671698.

[24] D. Lee, T. Ryan, H. J. Kim, Autonomous landing of a VTOL UAV on a moving platform using image-based visual servoing, in: 2012 IEEE International Conference on Robotics and Automation, IEEE, 2012. doi:10.1109/icra.2012.6224828.

[25] N. Guenard, T. Hamel, R. Mahony, A practical visual servo control for an unmanned aerial vehicle, IEEE Transactions on Robotics 24 (2) (2008) 331–340. doi:10.1109/tro.2008.916666.

[26] H. Lee, S. Jung, D. H. Shim, Vision-based UAV landing on the moving vehicle, in: 2016 International Conference on Unmanned Aircraft Systems (ICUAS), IEEE, 2016. doi:10.1109/icuas.2016.7502574.

[27] M. Fu, K. Zhang, Y. Yi, C. Shi, Autonomous landing of a quadrotor on an UGV, in: 2016 IEEE International Conference on Mechatronics and Automation, IEEE, 2016. doi:10.1109/icma.2016.7558697.

[28] J. L. Sanchez-Lopez, J. Pestana, S. Saripalli, P. Campoy, An approach toward visual autonomous ship board landing of a VTOL UAV, Journal of Intelligent & Robotic Systems 74 (1-2) (2013) 113–127. doi:10.1007/s10846-013-9926-3.

[29] O. Araar, N. Aouf, I. Vitanov, Vision based autonomous landing of multi-rotor UAV on moving platform, Journal of Intelligent & Robotic Systems 85 (2) (2016) 369–384. doi:10.1007/s10846-016-0399-z.

[30] K. E. Wenzel, A. Masselli, A. Zell, Automatic take off, tracking and landing of a miniature UAV on a moving carrier vehicle, Journal of Intelligent & Robotic Systems 61 (1-4) (2010) 221–238. doi:10.1007/s10846-010-9473-0.

[31] J. S. Wynn, T. W. McLain, Visual servoing with feed-forward for precision shipboard landing of an autonomous multirotor, in: 2019 American Control Conference (ACC), IEEE, 2019, pp. 3928–3935. doi:10.23919/acc.2019.8814694.

[32] G. Cho, J. Choi, G. Bae, H. Oh, Autonomous ship deck landing of a quadrotor UAV using feed-forward image-based visual servoing, Aerospace Science and Technology 130 (2022) 107869. doi:10.1016/j.ast.2022.107869.

[33] H. Romero, S. Salazar, O. Santos, R. Lozano, Visual odometry for autonomous outdoor flight of a quadrotor UAV, in: 2013 International Conference on Unmanned Aircraft Systems (ICUAS), IEEE, 2013. doi:10.1109/icuas.2013.6564748.

[34] R. Strydom, S. Thurrowgood, M. V. Srinivasan, Visual odometry: autonomous uav navigation using optic flow and stereo, in: Proceedings of Australasian conference on robotics and automation, 2014.

[35] T. Mouats, N. Aouf, L. Chermak, M. A. Richardson, Thermal stereo odometry for UAVs, IEEE Sensors Journal 15 (11) (2015) 6335–6347. doi:10.1109/jsen.2015.2456337.

[36] Y. Lu, Z. Xue, G.-S. Xia, L. Zhang, A survey on vision-based UAV navigation, Geo-spatial Information Science 21 (1) (2018) 21–32. doi:10.1080/10095020.2017.1420509.

[37] G. Balamurugan, J. Valarmathi, V. P. S. Naidu, Survey on UAV navigation in GPS denied environments, in: 2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES), IEEE, 2016. doi:10.1109/scopes.2016.7955787.

[38] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, P. Furgale, Keyframe-based visual–inertial odometry using nonlinear optimization, The International Journal of Robotics Research 34 (3) (2014) 314–334. doi:10.1177/0278364914554813.

[39] K. Schmid, P. Lutz, T. Tomić, E. Mair, H. Hirschmüller, Autonomous vision-based micro air vehicle for indoor and outdoor navigation, Journal of Field Robotics 31 (4) (2014) 537–570. doi:10.1002/rob.21506.

[40] Y. Liu, R. Xiong, Y. Wang, H. Huang, X. Xie, X. Liu, G. Zhang, Stereo visual-inertial odometry with multiple Kalman filters ensemble, IEEE Transactions on Industrial Electronics 63 (10) (2016) 6205–6216. doi:10.1109/tie.2016.2573765.

[41] J.-P. Tardif, M. George, M. Laverne, A. Kelly, A. Stentz, A new approach to vision-aided inertial navigation, in: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2010. doi:10.1109/iros.2010.5651059.

[42] M. Bloesch, M. Burri, S. Omari, M. Hutter, R. Siegwart, Iterated extended Kalman filter based visual-inertial odometry using direct photometric feedback, The International Journal of Robotics Research 36 (10) (2017) 1053–1072. doi:10.1177/0278364917728574.

[43] A. I. Mourikis, S. I. Roumeliotis, A multi-state constraint Kalman filter for vision-aided inertial navigation, in: Proceedings 2007 IEEE International Conference on Robotics and Automation, IEEE, 2007. doi:10.1109/robot.2007.364024.

[44] C. Forster, L. Carlone, F. Dellaert, D. Scaramuzza, On-manifold preintegration for real-time visual–inertial odometry, IEEE Transactions on Robotics 33 (1) (2017) 1–21. doi:10.1109/tro.2016.2597321.

[45] P. M. Gupta, E. Pairet, T. Nascimento, M. Saska, Landing a UAV in harsh winds and turbulent open waters, IEEE Robotics and Automation Letters 8 (2) (2023) 744–751. doi:10.1109/lra.2022.3231831.

[46] C. Kim, E. M. Lee, J. Choi, J. Jeon, S. Kim, H. Myung, ROLAND: Robust landing of UAV on moving platform using object detection and UWB based extended kalman filter, in: 2021 21st International Conference on Control Automation and Systems (ICCAS), IEEE, 2021. doi:10.23919/iccas52745.2021.9649920.

[47] K. Xia, M. Shin, W. Chung, M. Kim, S. Lee, H. Son, Landing a quadrotor UAV on a moving platform with sway motion using robust control, Control Engineering Practice 128 (2022) 105288. doi:10.1016/j.conengprac.2022.105288.

[48] L. Ambroziak, M. Ciężkowski, A. Wolniakowski, S. Romaniuk, A. Bożko, D. Ołdziej, C. Kownacki, Experimental tests of hybrid VTOL unmanned aerial vehicle designed for surveillance missions and operations in maritime conditions from ship-based helipads, Journal of Field Robotics 39 (3) (2021) 203–217. doi:10.1002/rob.22046.

[49] Y. Kang, B.-J. Park, A. Cho, C.-S. Yoo, Y. Kim, S. Choi, S.-O. Koo, S. Oh, A precision landing test on motion platform and shipboard of a tilt-rotor UAV based on RTK-GNSS, International Journal of Aeronautical and Space Sciences 19 (4) (2018) 994–1005. doi:10.1007/s42405-018-0081-8.

[50] K. Gamagedara, T. Lee, Geometric adaptive controls of a quadrotor unmanned aerial vehicle with decoupled attitude dynamics, Journal of Dynamic Systems, Measurement, and Control 144 (3) (Nov. 2021). doi:10.1115/1.4052714.

[51] K. Gamagedara, T. Lee, M. Snyder, Quadrotor state estimation with IMU and delayed real-time kinematic GPS, IEEE Transactions on Aerospace and Electronic Systems 57 (5) (2021) 2661–2673. doi:10.1109/taes.2021.3061795.

[52] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, D. Scaramuzza, SVO: Semidirect visual odometry for monocular and multicamera systems, IEEE Transactions on Robotics 33 (2) (2017) 249–265. doi:10.1109/tro.2016.2623335.

[53] Robotics and Perception Group at University of Zurich, RPG SVO Pro library, GitHub: `https://github.com/uzh-rpg/rpg_svo_pro_open`, accessed: 2022-July (2022).

[54] W. L. Brogan, Modern control theory, Prentice Hall, 1991.

[55] E. Hairer, M. Hochbruck, A. Iserles, C. Lubich, Geometric numerical integration, Oberwolfach Reports (2006) 805–882doi:10.4171/owr/2006/14.

[56] VectorNav, VN100 inertial measurement unit, `https://www.vectornav.com/`, accessed: 2023-Jan (2017).

[57] Swift Navigation, Piksi multi GNSS module, `https://www.swiftnav.com/`, accessed: 2023-Jan (2018).

[58] FreeWave, Freewave MM2-T telemetry module, `https://www.freewave.com`, accessed: 2023-Jan (2018).

[59] FLIR, Chameleon3 USB3, `https://www.flir.com/`, accessed: 2023-Jan (2018).

[60] T-Motor, U3 KV700, `https://store.tmotor.com`, accessed: 2023-Jan (2019).

[61] MikroKopter, BL-Ctrl, `https://wiki.mikrokopter.de`, accessed: 2023-Jan (2019).

[62] The GNOME Project, gtkmm: C++ Interfaces for GTK and GNOME, Website: `https://gtkmm.org`, accessed: 2023-Jan (Oct. 2021).

[63] Flight Dynamics & Control Lab at George Washington University, Autonomous shipboard landing of a UAV, YouTube: `https://youtu.be/o3fbh8TyZOs`, accessed: 2023-Jan (Oct. 2021).

[64] Flight Dynamics & Control Lab at George Washington University, Autonomous shipboard landing of a UAV in a GPS denied environment, YouTube: `https://youtu.be/v8euxR5UpUY`, accessed: 2023-Jan (Mar. 2022).

[65] C. Doer, G. F. Trommer, Radar visual inertial odometry and radar thermal inertial odometry: Robust navigation even in challenging visual conditions, in: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2021, pp. 331–338.