2020 IEEE International Symposium on
Safety, Security, and Rescue Robotics (SSRR)
November 4-6, Abu Dhabi, UAE

Autonomous Landing of a UAV on a Moving Ground Vehicle in a GPS Denied Environment

Stephen M. Nogar[1]

Abstract— **Teams of autonomous air and ground agents can enable operations in search and rescue environments that are too remote or dangerous for humans but often require extensive positioning and communication infrastructure. A straightforward approach that maximizes the use of modern robotics tools is implemented to achieve a rotary wing unmanned air vehicle (UAV) landing on a moving unmanned ground vehicle (UGV) without the aid of GPS or other external positioning systems. Landing uses positioning information from a "fractal" visual fiducial marker that embeds smaller markers within larger ones that allows the estimation of a 6 degree-of-freedom (DOF) pose at all altitudes during landing. All computations are performed onboard the vehicle using low cost sensors and computers and no communications are required between the UAV and UGV. Trials were conducted in both software-in-the-loop (SITL) simulations and outdoor experiments and demonstrated this algorithm to be effective at performing landing maneuvers. Landing was performed using a small custom-built quadrotor and a Clearpath Warthog UGV at speeds of up to 2.5 m/s.**

I. Introduction

In public safety and disaster scenarios, it is often too dangerous for humans to safely perform operations. In these scenarios, it is beneficial to have teams of autonomous robots enter a high risk area to gather information or perform rescue operations [1], [2]. Small, autonomous unmanned aerial vehicles (UAVs) are an enabling technology that offer increased detection and logistics support during these events where situational awareness may be limited [3]. Traditionally, autonomous agents require a positioning source such as GPS to navigate the environment along with communications infrastructure to coordinate between agents and operators.

However, in search and rescue scenarios GPS and communication infrastructure may not to be widely available or reliable due to urban structures blocking signals or damaged equipment [4]. Reducing or eliminating the need for GPS and communications would enable behaviors such as collaborative search, or the ability for the UAV to land on the UGV to recharge. Particularly, landing on a moving vehicle would provide the ability for autonomous agents to perform long duration missions without the need for human intervention. This work proposes and demonstrates a framework for landing an autonomous UAV on a moving ground vehicle without requiring an external positioning source such as GPS or communication between the vehicles.

In recent decades, multiple developments have coalesced to make these types of operations feasible and simpler to implement. Advances in embedded computing have made it possible to estimate position based on fusing visual and inertial data onboard the aircraft [5], [6]. Lightweight Lidar [7], depth sensors [8], embedded cameras [9], rangefinders [10], and ultra-wide band sensors [11] have also been developed to aid in positioning and localizing objects in the environment. These sensors, combined with low cost flight controllers, such as the Pixhawk [12], along with supporting software such as the PX4 flight control and Robot Operating System (ROS) have made the development of GPS denied behaviors feasible and practical.

Landing on a moving vehicle has been previously accomplished using a variety of methods. Its common to use a traditional camera with a fiducial marker on the landing pad to use as a visual servoing target [13], [14] or to estimate a 6 degree of freedom (DoF) pose [15], [16]. More recent work has included using deep neural networks (DNN) to improve landing control [17], [18] and fiducial marker detection [19]. Often, these implementations rely on external positioning [20]–[22], computing offboard the UAV [23], or inertial data from the ground vehicle [21], limiting their utility in search and rescue environments. An alternative to fiducial markers can be active beacons placed on the landing pad, such as infra-red [24], or ultra-wideband [25]. However, these systems require additional energy to operate, and also add additional hardware on the UAV, increasing weight.

A common issue with fiducial marker based approaches is that they often rely on a single marker that must be visible both at high altitude and close to the landing pad, requiring evaluating a complex set of trade-offs of camera resolution, onboard processing power, and suitable detection ranges. If a marker is large enough to be detected from high altitude, it may become unobservable at low altitudes. The work of [26]–[28] addresses this problem by is adopting a "fractal" version of an Aruco marker that embeds smaller markers within larger markers making it observable over a wide variety of heights while also maintaining computational tractability that can run at high rates on constrained hardware.

This work seeks to implement a straightforward approach that maximizes use of widely available flight control, sensing, and computing platforms to land on a moving ground vehicle without dependence on GPS or communication with the UGV. It leverages open source software frameworks such as ROS, Gazebo and PX4 to minimize implementation time, and relies on the computationally lightweight fractal Aruco markers to provide positioning information at a high rate to enable landing in an outdoor environment at speeds of approximately 2.5 m/s. A software in the loop (SITL) testing

This work was supported by the U.S. Army Research Laboratory
[1]Aerospace Engineer, Army Research Laboratory, Aberdeen Proving Ground, MD, 21005, USA, stephen.m.nogar.civ@mail.mil

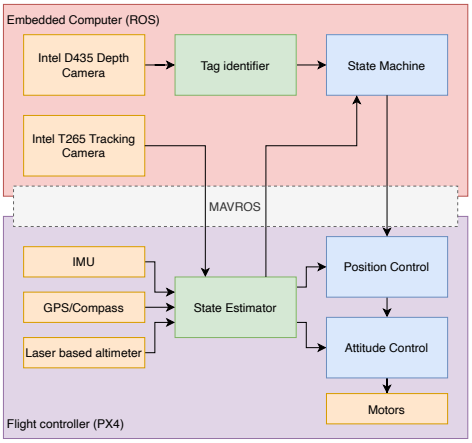

Fig. 1. UAV software architecture with PX4 and ROS components.

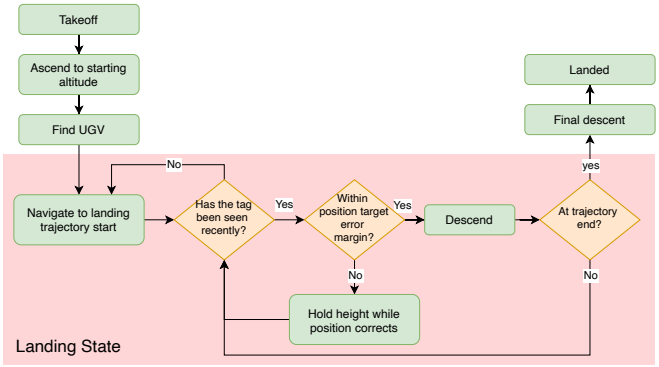

Fig. 2. State machine and landing process diagram.

environment enables quick prototyping and allowed outdoor experiments to be completed with only a limited number of trials.

II. Implementation

The objective of this implementation is to maximize the use of available open-source software, commodity sensors, and compute hardware to create an easy to implement landing model. The system architecture and information flow is illustrated in Fig. 1. The key contributions are the Aruco marker tag identifier, and the state machine which includes the trajectory planner and moving landing position controller.

The mission is pre-programmed to takeoff, find the landing pad, and perform the landing itself. Practical considerations such as robust vision based localization, obstacle avoidance, and the ability to search for the ground vehicle if its location is unknown are not addressed in this work.

A. State Machine

A finite state machine was developed to provide high-level control to the vehicle during each phase of the flight. Illustrated in Fig. 2, each state performs the requested action and proceeds to the following state. Implemented using the Smach library within ROS, this system provides a robust interface for building complex behaviors.

States such as “takeoff” and “ascend to starting altitude” simply assign a target position setpoint and insure that the vehicle has achieved the setpoint before proceeding to the next state. The “Find UGV” state navigates to a pre-assigned point while also waiting for the marker on the unmanned ground vehicle (UGV) to be detected. Once a marker detection occurs, the landing sequence begins. The landing controller implementation described below is contained within “Landing State” and is thus also implemented in the Smach framework.

B. Fractal Fiducial Marker

A fractal Aruco marker [26]–[28] is used to provide 6 DoF localization capabilities at a wide range of heights above the marker. The fractal algorithm builds on the speed and pose estimation capabilities of standard Aruco markers [29], commonly used in augmented and virtual reality applications. Within this work, the fractal marker algorithm was adapted to run within ROS. The algorithm is able to track all detectable fractal markers within an image, but only returns a single pose estimate, keeping computational cost low. Marker detections could be computed at a rate of 90 Hz on a single core of the Intel Core i7 CPU running onboard the vehicle.

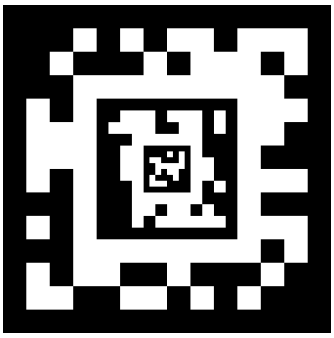

Fig. 3. Selected fractal Aruco marker.

The fractal marker can be configured to have many different combinations of the number of embedded markers and the size and complexity of the pixels within each marker depending on the desired characteristics. Three layers of markers were selected due to the wide range of expected altitudes and relatively narrow lens of the selected camera. The marker used in this study is shown in Fig. 3.

C. Landing Algorithm

The landing sequence depends on several components; detection of the marker and transforming it into a reference frame that can be used for control, trajectory planning, and a controller that assumes that the landing pad reference frame is moving. While vision based positioning could aid the landing process, qualitative experiments with the Intel T265 tracking camera without the UGV present found that often tracking was lost in flight tests in a grassy field. Even if positioning were available, the sensor would likely be

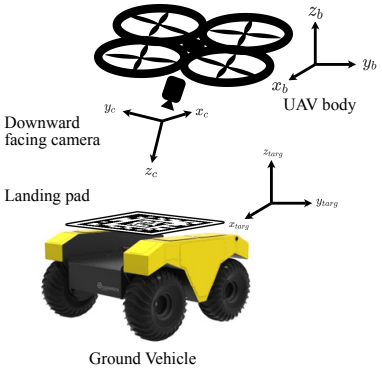

Fig. 4. UAV and ground vehicle coordinate frames.

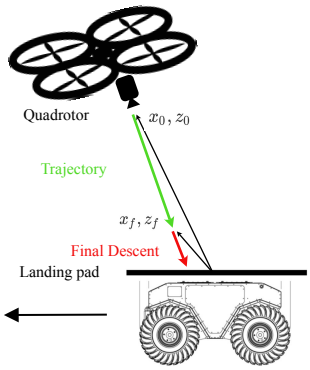

Fig. 5. Landing path diagram. Coordinates are defined in the pitch-roll corrected frame fixed to the landing pad.

obscured by the UGV during landing. Therefore, positioning information relative to the ground from sensors such as VIO or GPS is assumed to be unavailable during the landing phase of the flight.

1) Pose Estimation: An important ability is to localize the control of the UAV in a coordinate frame that is fixed to the landing pad on the ground vehicle. Coordinate frames for the UAV, camera, and landing pad are shown in Fig. 4. The body frames of the UAV and the UGV are assumed to be in the east-north-up (ENU) coordinate frame, while the camera frame is aligned with z pointing out of the camera lens. Forward for the UGV is assumed to be in the positive x direction.

When the marker is identified, the pose of the UAS is estimated as:

$$\boldsymbol{T}^b_{pad} = \boldsymbol{T}^b_c \boldsymbol{T}^c_{pad} \tag{1}$$

Where b is the UAV body frame in ENU coordinates, c is the camera frame and pad is the landing pad coordinates.

The target UAV trajectory is determined relative to the landing pad, which could be subject to high-levels of noise due to the interactions of the UGV with uneven terrain. Additionally, if the UGV is on sloped ground, a constant pitch or roll of the landing pad would be present which should not be factored into the landing trajectory. When projecting the trajectory from the pad these errors are amplified, especially for the start of the trajectory. A series of transformations are defined to remove the rolling and pitching of the landing pad:

$$\begin{aligned} \boldsymbol{T}^{b_{att}}_{pad} &= \boldsymbol{T}^n_{b_{att}} \boldsymbol{T}^b_{pad} \\ \tilde{\boldsymbol{T}}^{b_{att}}_{pad} &= \boldsymbol{T}^{b_{att}}_{pad} \tilde{\boldsymbol{T}}_{b_{att}} \\ \tilde{\boldsymbol{T}}^b_{pad} &= \boldsymbol{T}^{b_{att}}_{pad}{}^{-1} \tilde{\boldsymbol{T}}^{b_{att}}_{pad} \end{aligned} \tag{2}$$

Where $T^n_{b_{att}}$ is the attitude of the vehicle, $\tilde{T}_{b_{att}}$ contains the negative roll and pitch components of $T^{b_{att}}_{pad}$ such that roll and pitch motions are canceled but yaw is still tracked. Sensors such as the accelerometer, gyroscope, and magnetometer allows the vehicle to track the horizon to perform these transformations, even when no position information is available resulting in a frame that is level with the horizon.

The desired relationship of the UAV relative to the landing pad is then found from

$$\tilde{\boldsymbol{T}}^{pad}_b = \tilde{\boldsymbol{T}}^b_{pad}{}^{-1} \tag{3}$$

and provides the reference frame fixed to the landing pad that can now be used to control the position of the UAV.

2) Path Planning: A planner was developed to provide a target trajectory for the vehicle to follow during landing. The trajectory is defined relative to the landing pad, and is shown in Fig. 5. The downward camera is rigidly mounted to the bottom of the UAV and to keep the marker in view while the vehicle is pitched in forward flight, the trajectory specifies that the UAV fly in front of the UGV to maintain visibility of the marker.

The flight path is defined relative to the center of the fiducial marker. Parameters x_0, z_0, x_f, z_f are defined in meters. Starting position x_0 represents the starting position in front of the vehicle and is scaled relative to the detected velocity of the ground vehicle using:

$$\begin{aligned} x_0 &= v_{g_{est}} x_{0_{\text{offset}}} \\ v_{g_{est}} &= \int_0^t \boldsymbol{E}_p dt \end{aligned} \tag{4}$$

The value of $x_{0_{\text{offset}}}$ is defined in m/(m/s), effectively changing the approach angle by the detected velocity. Value $v_{g_{est}}$ is the integral term computed by the controller to account for the constant movement of the ground vehicle.

During landing, the vehicle descends at a constant velocity. If the vehicle deviates from the trajectory at a magnitude greater than the acceptance sphere, the descent is paused until it reenters the acceptance sphere. If the marker is not detected for a sufficient amount of time, the landing process is restarted, and the initial height of z_0 is set while the controller waits for the marker to be reacquired. This sequence of events is illustrated in Fig. 2.

Once the UAV descends to the final point on the trajectory at z_f, a final descent stage is initiated that ends a constant throttle value, less than that needed to hover, along with a level attitude command for a short, predetermined amount of time. Then the motors are disarmed.

3) Control: A simple proportional-derivative-integral controller is proposed here to achieve the landing. Since no absolute position or velocity reference is available, the integral term is heavily relied upon to account for the steady motion of the landing pad. Position error is determined with

$$\boldsymbol{E}_p = \tilde{\boldsymbol{T}}^{pad}_{b_{targ}} \quad \tilde{\boldsymbol{T}}^{pad}_{b} \tag{5}$$

Term $\boldsymbol{T}^{pad}_{b_{targ}}$ is composed of the current target coordinates in the landing trajectory as defined in the previous section. Term $\boldsymbol{E}_p$ includes a saturation limit to prevent high error values causing dramatic motions of the UAV before it is has reached the trajectory start point.

The target velocity is determined by:

$$\boldsymbol{V}^{pad}_{b} = \boldsymbol{E}_p\boldsymbol{K}_p + \boldsymbol{K}_I \int_0^t \boldsymbol{E}_p\boldsymbol{K}_p dt \quad \boldsymbol{v}_{pad}\boldsymbol{K}_D + V_{ff} \tag{6}$$

Where $\boldsymbol{K}_P$, $\boldsymbol{K}_I$, $\boldsymbol{K}_D$ are the position, integral, and derivative terms. The feedforward term $\boldsymbol{V}_{ff}$ is simply the descent velocity. Term $\boldsymbol{v}_{pad}$ is the velocity of UAV relative to the landing pad and the derivative is computed numerically using a moving average filter. No filtering is applied to position and yaw.

The velocity in Eq. (6) is rotated to the inertial frame using:

$$\boldsymbol{V}^n = \boldsymbol{T}^n_b\boldsymbol{T}^b_{pad}\boldsymbol{V}^{pad}_b \tag{7}$$

Eq. 7 is input into the flight controller, which even in the absence of position data uses pitch and roll angles of the UAV to approximate velocity. The target yaw is also included that corresponds to the detected direction of the landing pad. This study leverages the PX4 flight control software, which uses uses a cascaded control design that uses a position controller to set a target attitude to reduce system error.

III. Experimental Setup

A UAV was built to perform the landing task and a list of important components is available in Table I. The Intel Realsense D435 depth camera was rigidly mounted on the bottom of the vehicle. Depth functionality of the camera was not used this experiment, only a single infrared imager running at a resolution of 640x480 at 90 Hz. The Intel Realsense T265 tracking camera was mounted to the front of the vehicle at a 45 degree angle below the horizon. The frame was constructed out of a carbon plate and the entire vehicle weighs approximately 1 kg without the battery. While a GPS is included on the vehicle, it was not enabled during landing attempts.

The software organization on the vehicle is outlined in Fig. 1. The landing algorithm is implemented in the ROS framework and is used to send both positioning information and control commands to the PX4 flight controller. Vision based position data is forwarded from the T265 tracking camera into the PX4 state estimator. Control commands are generated from the state machine and are used as position and attitude control setpoints depending on the flight mode. The flight controller provides sensor data, fused localization information and various other status updates about the vehicle during flight. ROS and PX4 information is translated by the MAVROS package that forward ROS data to the flight controller over a Mavlink serial link.

TABLE I
List of parts included on the vehicle.

Component	Model
Flight Controller	Pixracer
Computer	4 core Intel Core i7 NUC
Tracking Camera	Intel Realsense T265
Marker Detection Camera	Intel Realsense D435
Rangefinder	LIDAR-Lite v3HP
GPS/Compass	mRo GPS u-Blox Neo-M8N
Motors	T-Motor F40
Propellers	3 bladed 5 inch radius

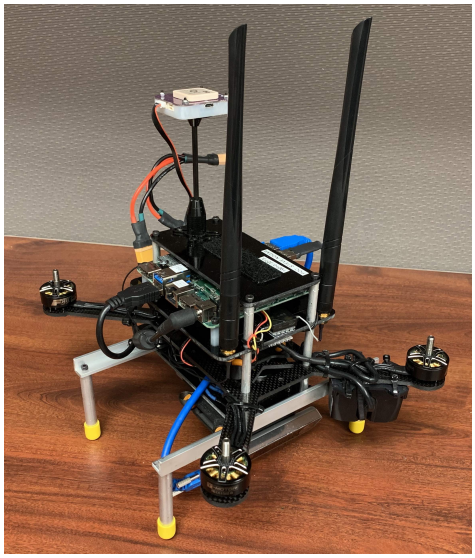

Fig. 6. Vehicle used for landing with propellers removed.

A Clearpath Warthog autonomous ground vehicle was used as the landing target. These vehicles have a maximum speed of 5 m/s, in this case it was remotely driven at half throttle at approximately 2.5 m/s. A landing pad was constructed that contained the fractal Aruco marker printed on matte paper to reduce reflections on top of the vehicle. The Aruco marker is sized at 0.635 m in length.

Software-in-the-loop based simulation testing was also performed before testing outdoors. These were performed on an 18 Core Intel Xeon Processor with AMD Radeon Pro Vega 64 graphics card. All software including the simulator, ROS based software, and PX4 flight controller were run simultaneously on that same computer.

IV. Results

The algorithm outlined in this work was demonstrated both in simulation and experimentally. The purpose of the simulations is to prototype the vision and control algorithms to reduce experimentation time.

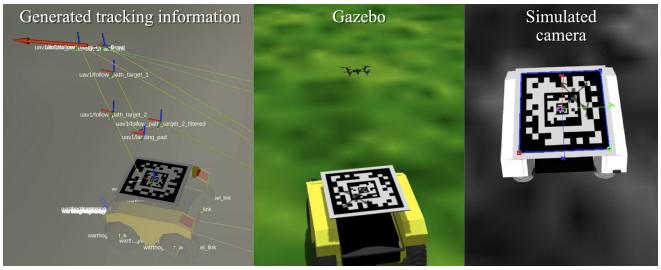

Fig. 7. Left: Data and coordinate frame visualization during landing, Middle: Gazebo simulation view, Right: Simulated camera view with marker detection overlaid.

A. Simulation

Software in the loop simulations are a valuable tool for quickly and efficiently testing new autonomous algorithms. In these tests, the majority of the autonomous software including the flight controller is kept the same, while hardware such as motors and sensors are substituted for simulated equivalents. In this case, the Gazebo simulator was used along with the ROS version of the RotorS plugins [30] to connect the PX4 flight controller to the simulator. A simulated Warthog was available from Clearpath and was modified to include the fractal Aruco marker. Both the ROS and the PX4 based software were compiled for use in a desktop environment. The proprietary implementation of the visual-inertial odometry (VIO) algorithm embedded in the Intel T265 tracking camera could not be replicated in simulation, and instead was substituted with ground truth data.

Simulated experiments proved to be extremely useful in the development of the above algorithm. Once the algorithm was complete, the vehicle was able to perform landings consistently. Fig. 7 shows views from the simulated landing that illustrates the capabilities of the environment including the ability to localize the Aruco marker. A surprising issue during testing was the UAV's shadow would sometimes obscure the marker, preventing landing in some cases. The sun was simulated approximately as it would be around the middle of the day on a cloudless day. Its unclear if these high contrast shadows are representative of reality or if they are artifacts of the Gazebo graphical engine. This behavior was not observed in outdoor tests.

An interesting behavior observed in simulation was the response to a lost marker event, such as when the UGV starts to drive forward and leaves the camera field of view. When this happens, the UAV ascends to z_0, and the control law in Eq. (6) continues to operate with the last known detected marker pose. The vehicle was observed to accelerate in the direction of the last known marker position relative to the camera. Since the UGV was traveling in a straight line, eventually the UAV overtakes the UGV and can restart the landing process with some degree of robustness. The integrator in Eq. (6) is primarily responsible for this behavior, which in the case of a large position error for a sustained amount of time quickly saturates. While not a comprehensive solution to searching for the UGV, this was a surprising result of the simple controller. However, due to safety and field length constraints, it was not validated experimentally.

B. Experiment

Experimental landings were performed at Aberdeen Proving Ground in Maryland. Only 6 days of outdoor testing was needed to achieve consistent landings. The UGV was manually driven in a straight line over a flat, grassy field. Snapshots from a trial run are shown in Fig. 8 including the processed camera data that tracks the fiducial marker. Vehicle pose and trajectory data are shown in Fig. 9. The UAV was commanded to takeoff and hover over the UGV. Once hovering, the UAV was commanded to search for the landing pad at the same time the Warthog began to move forward.

The efficacy of algorithm was validated through several landing attempts ranging in speeds up to 2.5 m/s. Landing performance was found to be sensitive to tuning values depending on the speed of the UGV. This was likely caused by the simplistic PID controller implementation. Tracking in x exhibits underdamped behavior, due to the difficulty of balancing position and integral terms to account for the forward motion of the UGV while also centering the UAV over the landing pad. Large position errors after the ground vehicle starts moving provide the initial excitation. Descent is paused from 44-49 s, and 50.5-52 s due to large errors in especially the x direction. Position in z is tracked well, since it aligns with the primary rotor thrust direction and the state estimator is able to achieve a high degree of accuracy from the combined barometer and rangefinder sensors.

During landing, the fractal Aruco marker was tracked well during all phases of the flight. Fig. 10 shows the detected position of the marker relative to the camera throughout the flight. In the initial hover phase of the flight from 35 to 39 s, some noise and dropouts are observed in the landing pad measurements, likely due to the limited camera resolution. Note that while the marker was always tracked, it was not used as a positioning source until the landing phase of the flight. As the UAV approached the pad, the tracking algorithm successfully switched from the larger marker to the inner markers, seen in the frame at 54.7 s of Fig. 8, demonstrating the clear benefits of the fractal marker for this application.

Initial positioning of the vehicle above the UGV was performed using the combined input of the T265 tracking camera, downward facing rangefinder, and the IMU+barometer embedded in the flight controller. The VIO algorithm in the T265 sometimes struggled to maintain effectiveness during initial positioning, and was very unreliable during the landing maneuver. Future work could refine the use of this sensor or other VIO algorithms to increase performance, and implement the ability to identify and mask the UGV during landing such that it does not inhibit tracking visual features.

Two factors appear to limit how fast landing can be performed. First, the simplistic controller requires retuning at different speeds, which limits its utility if the speed is

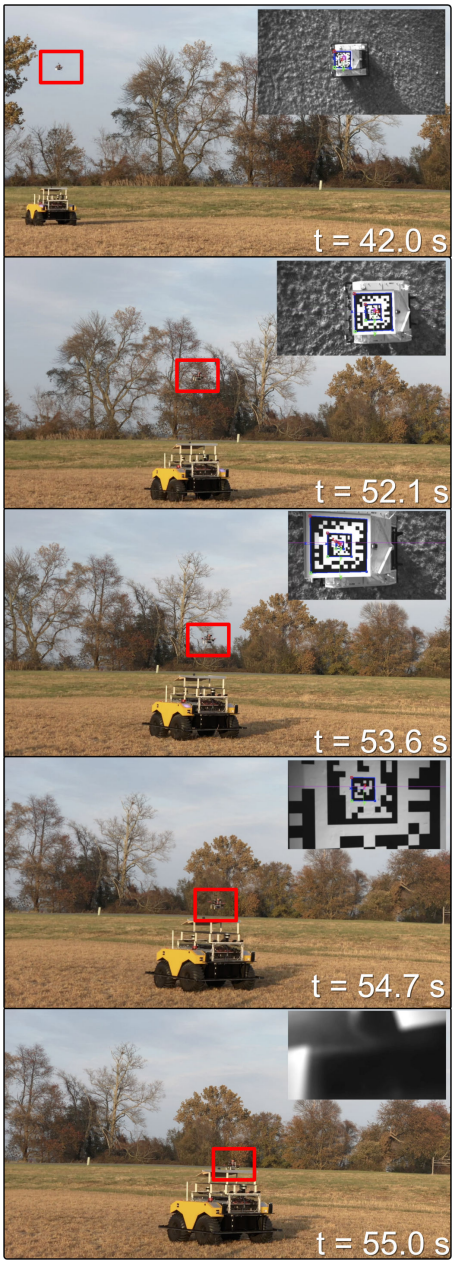

Fig. 8. Landing sequence. The UAV is highlighted in the red box. The top right square shows the processed camera frame with the fiducial marker label overlaid.

unknown a priori. Future work could implement a gain scheduling scheme, or implement a generally more robust control scheme. A second limiting factor is as the UGV speed increases, the UAV must make more aggressive movements to perform the landing and with the camera rigidly mounted to the vehicle, keeping the marker in frame becomes challenging. This could be alleviated in several different ways. First, the camera could be mounted on a gimbal designed to track the UGV during the landing. Second, a more robust UGV tracking scheme could be designed that would predict the location of the UGV after it has left the camera frame,

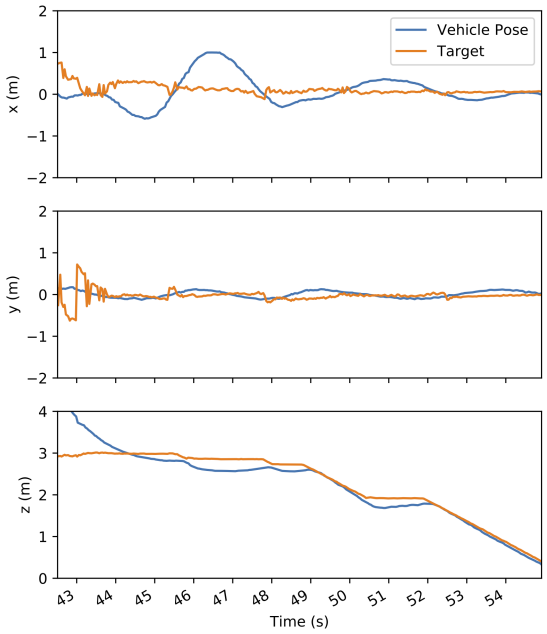

Fig. 9. Vehicle position and target trajectory relative to the landing pad during final landing phase.

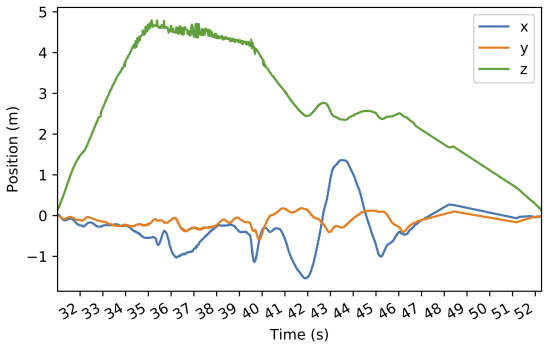

Fig. 10. Aruco marker detection position throughout the entire flight.

allowing the UAV to have a higher probability of reacquiring the UGV as it makes corrective maneuvers. UGV tracking would be significantly aided by a robust visual positioning system that would provide a ground reference frame. Finally, the UGV and UAV could share any available inertial or positioning information, adding significant aid to the UAV predicting current and future movements of the UGV.

V. Conclusions and Future Work

This work implements a straightforward algorithm that maximizes modern robotics tools and low cost UAV hardware and sensors. The result is a modular framework for performing landing using a small UAV at speeds up to 2.5 m/s. The fractal Aruco marker is utilized effectively and was found to be very well suited to this task. Simulation based testing demonstrated the algorithm was able to reacquire and restart the landing process in some cases of the marker being

lost from the camera view.

Software-in-the-loop simulations proved to be extremely valuable and minimized the experimental testing time necessary. Test flights were only necessary for 6 days before robust landing performance was achieved, validating the excellent work of [30] and others who contribute to the PX4, ROS, and Gazebo ecosystem. The ability to run the same code both in simulation and in experiment is an invaluable tool to the robotics researcher and can reduce the often destructive nature of the UAV build-test-fly iteration loop.

Future work could include the addition of robust control approaches, and a more robust treatment of vision based navigation such as more robust methods to search and localize the ground vehicle and improved visual-inertial odometry to maintain ground-relative localization during landing. These improvements should allow landing algorithms such as this one to operate more robustly, at higher speeds, and in a wider variety of environments to enable effective air-ground teaming in search and rescue environments.

ACKNOWLEDGMENT

The author would like to thank Leonard Bean, Howard Carpenter, Levi Carr, Jim Dotterweich, and Will Jones for fabrication and experimental assistance.

REFERENCES

[1] Tomic, T., Schmid, K., Lutz, P., Domel, A., Kassecker, M., Mair, E., Grixa, I. L., Ruess, F., Suppa, M., and Burschka, D., "Toward a Fully Autonomous UAV: Research Platform for Indoor and Outdoor Urban Search and Rescue," *IEEE Robotics Automation Magazine*, Vol. 19, No. 3, 2012, pp. 46–56.

[2] Michael, N., Shen, S., Mohta, K., Kumar, V., Nagatani, K., Okada, Y., Kiribayashi, S., Otake, K., Yoshida, K., Ohno, K., et al., "Collaborative mapping of an earthquake damaged building via ground and aerial robots," *Field and service robotics*, Springer, 2014, pp. 33–47.

[3] Kumar, V. and Michael, N., "Opportunities and challenges with autonomous micro aerial vehicles," *The International Journal of Robotics Research*, Vol. 31, No. 11, 2012, pp. 1279–1291.

[4] Lee, E. S., Loianno, G., Thakur, D., and Kumar, V., "Experimental Evaluation and Characterization of Radioactive Source Effects on Robot Visual Localization and Mapping," *IEEE Robotics and Automation Letters*, Vol. 5, No. 2, 2020, pp. 3259–3266.

[5] Loianno, G., Watterson, M., and Kumar, V., "Visual inertial odometry for quadrotors on SE(3)," *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 1544–1551.

[6] Lee, S., Har, D., and Kum, D., "Drone-assisted disaster management: Finding victims via infrared camera and lidar sensor fusion," *2016 3rd Asia-Pacific World Congress on Computer Science and Engineering (APWC on CSE)*, IEEE, 2016, pp. 84–89.

[7] "OSO Lidar," https://ouster.com/products/os0-lidar-sensor/, Accessed: 2020-08-26.

[8] "Intel® RealSense™ Depth Camera D435," https://www.intelrealsense.com/depth-camera-d435/, Accessed: 2020-08-26.

[9] "Qualcomm® Snapdragon Flight™ Kit," https://www.intrinsyc.com/vertical-development-platforms/qualcomm-snapdragon-flight/, Accessed: 2016-04-29.

[10] "LIDAR-Lite v3 Optical Distance Measurement Sensor," https://buy.garmin.com/en-US/US/p/557294, Accessed: 2020-08-26.

[11] "Loco Positioning system," https://www.bitcraze.io/products/loco-positioning-system/, Accessed: 2020-08-26.

[12] "Pixhawk Series," https://docs.px4.io/master/en/flight_controller/pixhawk_series.html, Accessed: 2020-08-26.

[13] Serra, P., Cunha, R., Hamel, T., Cabecinhas, D., and Silvestre, C., "Landing of a quadrotor on a moving target using dynamic image-based visual servo control," *IEEE Transactions on Robotics*, Vol. 32, No. 6, 2016, pp. 1524–1535.

[14] Lee, D., Ryan, T., and Kim, H. J., "Autonomous landing of a VTOL UAV on a moving platform using image-based visual servoing," *2012 IEEE international conference on robotics and automation*, IEEE, 2012, pp. 971–976.

[15] Saripalli, S., Montgomery, J. F., and Sukhatme, G. S., "Vision-based autonomous landing of an unmanned aerial vehicle," *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, Vol. 3, IEEE, 2002, pp. 2799–2804.

[16] Falanga, D., Zanchettin, A., Simovic, A., Delmerico, J., and Scaramuzza, D., "Vision-based autonomous quadrotor landing on a moving platform," *2017 IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, IEEE, 2017, pp. 200–207.

[17] Rodriguez-Ramos, A., Sampedro, C., Bavle, H., Moreno, I. G., and Campoy, P., "A Deep Reinforcement Learning Technique for Vision-Based Autonomous Multirotor Landing on a Moving Platform," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 1010–1017.

[18] Almeshal, A. M. and Alenezi, M. R., "A vision-based neural network controller for the autonomous landing of a quadrotor on moving targets," *Robotics*, Vol. 7, No. 4, 2018, pp. 71.

[19] Truong, N. Q., Nguyen, P. H., Nam, S. H., and Park, K. R., "Deep learning-based super-resolution reconstruction and marker detection for drone landing," *IEEE Access*, Vol. 7, 2019, pp. 61639–61655.

[20] Ghamry, K. A., Dong, Y., Kamel, M. A., and Zhang, Y., "Real-time autonomous take-off, tracking and landing of UAV on a moving UGV platform," *2016 24th Mediterranean conference on control and automation (MED)*, IEEE, 2016, pp. 1236–1241.

[21] Borowczyk, A., Nguyen, D.-T., Phu-Van Nguyen, A., Nguyen, D. Q., Saussié, D., and Le Ny, J., "Autonomous landing of a multirotor micro air vehicle on a high velocity ground vehicle," *Ifac-Papersonline*, Vol. 50, No. 1, 2017, pp. 10488–10494.

[22] Beul, M., Houben, S., Nieuwenhuisen, M., and Behnke, S., "Fast autonomous landing on a moving target at MBZIRC," *2017 European Conference on Mobile Robots (ECMR)*, IEEE, 2017, pp. 1–6.

[23] Lee, B., Saj, V., Benedict, M., and Kalathil, D., "A Vision-Based Control Method for Autonomous Landing of Vertical Flight Aircraft On a Moving Platform Without Using GPS," *arXiv preprint arXiv:2008.05699*, 2020.

[24] Wenzel, K. E., Masselli, A., and Zell, A., "Automatic Take Off, Tracking and Landing of a Miniature UAV on a Moving Carrier Vehicle," *Journal of Intelligent & Robotic Systems*, Vol. 61, No. 1-4, Oct. 2010, pp. 221–238.

[25] Nguyen, T. H., Cao, M., Nguyen, T., and Xie, L., "Post-Mission Autonomous Return and Precision Landing of UAV," *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, 2018, pp. 1747–1752.

[26] Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F., and Medina-Carnicer, R., "Generation of fiducial marker dictionaries using Mixed Integer Linear Programming," *Pattern Recognition*, Vol. 51, 10 2015.

[27] Romero-Ramirez, F., Muñoz-Salinas, R., and Medina-Carnicer, R., "Fractal Markers: a new approach for long-range marker pose estimation under occlusion," *IEEE Access*, Vol. PP, 11 2019, pp. 1–1.

[28] Romero-Ramirez, F., Muñoz-Salinas, R., and Medina-Carnicer, R., "Speeded Up Detection of Squared Fiducial Markers," *Image and Vision Computing*, Vol. 76, 06 2018.

[29] Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F. J., and Marín-Jiménez, M. J., "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, Vol. 47, No. 6, June 2014, pp. 2280–2292.

[30] Furrer, F., Burri, M., Achtelik, M., and Siegwart, R., *Robot Operating System (ROS): The Complete Reference (Volume 1)*, chap. RotorS—A Modular Gazebo MAV Simulator Framework, Springer International Publishing, Cham, 2016, pp. 595–625.