2023 IEEE 26th International Conference on
Intelligent Transportation Systems (ITSC)
24-28 September 2023. Bilbao, Bizkaia, Spain

Landing a Quadrotor on a Ground Vehicle without Exteroceptive Airborne Sensors: A Non-Robocentric Framework and Implementation*

Li-Yu Lo[1] *Student Member, IEEE*, Boyang Li[2], Chih-Yung Wen[1], and Ching-Wei Chang[1†]

Abstract— **This research addresses the problem of a quadrotor UAV landing on a ground vehicle. Yet, unlike most existing literature, we transfer most sensing and computing tasks to the ground vehicle, designing the landing system in a non-robocentric fashion. Such a framework greatly alleviates the payload burden, allowing more resource allocation for the quadrotor UAV. To validate the proposed framework, the implementation starts with relative pose estimation through detection and tracking of LED markers on an aerial vehicle. The 6 DoF orientation and position information is then returned through a PnP-based algorithm. Successively, by considering the visibility and dynamic constraints in the target local frame, the motion planning module computes an optimized landing trajectory, such that the aerial vehicle stays within a safety corridor and performs the landing mission. Through experiments, we demonstrate the applicability of this research work, in which a quadrotor could be guided remotely and landed on a moving ground vehicle smoothly without the support from any airborne exteroceptive sensors and computers.**

SUPPLEMENTARY MATERIAL

Supporting video of this paper is available at: `https://youtu.be/7wiCh46MQmc`

I. INTRODUCTION

Quadrotor UAVs possess high versatility and easy deployment. In a variety of scenarios, aerial robots have revealed their unprecedented applicability, including yet not limited to, precise agriculture [1], surveillance [2], [3], search and rescue [4], exploration and mapping [5], and so forth. Among all the aforementioned tasks, safe, autonomous, and efficient landing undoubtedly plays a crucial role. For instance, flight missions such as marine surveying take place in surroundings where the final retraction of UAV could affect the overall assignment [6]. Additionally, for quadrotors that are sent out to conduct suburban package shipments, taking-off from and landing on a constantly moving vehicle could be one application scenario [7]. Furthermore, it is well-known that rotary aircraft are prone to have short airborne time due to their low power efficiency, and therefore aerial robots frequently returning to their charging stations could be commonly seen in future smart city applications [8].

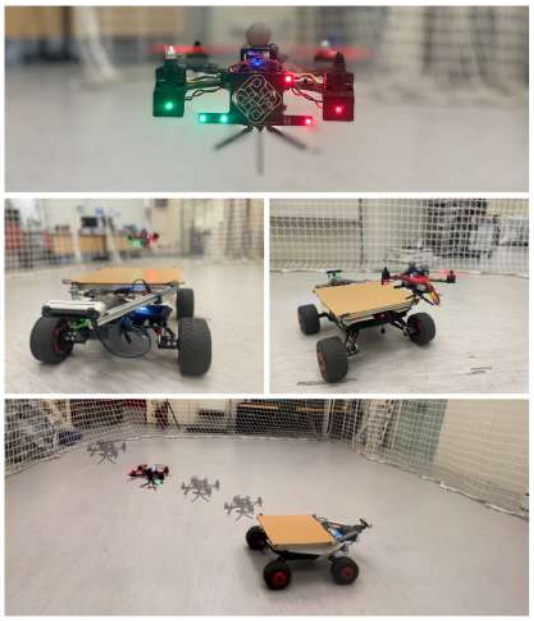

Fig. 1. The proposed non-robocentric landing framework and implementation. In which, the quadrotor, with no onboard exteroceptive sensors and computers, is passively guided to land on a dynamic platform

Nevertheless, the uncertainty of the landing environment usually poses great challenges to such operations, especially when final touchdown locations are in a dynamic motion such as on ground vehicles or surface vessels; this hence provides a worth-investigating research topic.

Yet, instead of approaching the problem in an onboard fashion, we transfer most sensing and computing modules to the ground, leaving the airborne quadrotor with solely a low-cost flight control unit, where we deem it a "non-robocentric" approach (shown in 1). Note that the term "non-robocentric" is defined with respect to the landing robot itself, where it possesses no independent sensing or computing capability for landing; in other words, all localization and control are done in a non-inertial frame. It is argued that, such a configuration could significantly alleviate the UAV payload and the onboard avionics system's complexity, while maintaining a similar performance to a conventional autonomous landing system. Furthermore, when a UAV mission does not require high computational perception modules (such as air quality monitoring presented in [6]), it is believed that the proposed system could act as a more optimal solution for retracting UAVs. To validate the proposed framework, an implementation is carried out, in which computer vision algorithms,

*This research is funded by the Innovation and Technology Commission of Hong Kong under grant number ITT/027/19GP and PolyU Presidential PhD Fellowship.

[1]Li-Yu Lo, Chih-Yung Wen and Ching-Wei Chang are with AIRo-LAB, Department of Aeronautical and Aviation Engineering, The Hong Kong Polytechnic University, Kowloon, Hong Kong (e-mail: patty.lo@connect.polyu.hk; chihyung.wen@polyu.edu.hk; ching-wei.chang@connect.polyu.hk).

[2]Boyang Li is with School of Engineering, The University of Newcastle, Callaghan, NSW 2308, Australia (e-mail: boyang.li@newcastle.edu.au).

[†] corresponding author

quadrotor motion planning, and control are harnessed and run on ground. To our best knowledge, this is the first complete demonstration of landing a quadrotor using a non-robocentric approach, where non-inertial localization and control are carried out.

A. Related Work

Autonomous landing for unmanned aerial systems has been addressed frequently during the past few years; meanwhile, advancements in low-cost computation components have also provided the field of machine vision a burgeoning opportunity to rise. The above two then led to a massive body of literature that dealt with both static and dynamic landing scenarios. One prominent early work was presented by Lee et al. [9], where a conventional visual servoing method to land a quadrotor on a moving platform was adopted. With a different strategy, Falanga et al. [10] applied state-of-the-art vision odometry [11] and trajectory planner [12] to design a complete quadcopter vision system that could perform landing maneuvers on a moving ground vehicle. With computer vision-based pattern recognition, the proposed algorithm was able to detect and track the relative state of the landing pad. Literature in [13], [14], [15], [16], [17] also exploited similar vision configurations to address the landing problem, but with different control methods under different scenarios.

Compared to onboard sensing and computing, the number of literature adopting ground-based guidance is relatively small due to its higher complexity when estimating the respective pose. Work presented in [18] manufactured a trinocular guidance system to land a helicopter. Through 3D reconstruction techniques, the position information of the aircraft could be acquired. With a similar idea, authors in [19], [20], proposed a pan-tilt stereo camera unit that could detect and track the UAV. Furthermore, to increase the robustness under all weather, the infrared spectrum was utilized. While the aforementioned works have shown great performance in dealing with static landing problems, Ferreira et al. [21] proposed a landing guidance system for dynamic situations, in which deep learning-based methods were used to recover the relative 6-DoF information between an aircraft carrier and a fixed-wing UAV. Nevertheless, the authors only validated the system through simulations, and therefore, further practical experiments might be required to prove its real-life applicability. Hence, this research aims to establish a general ground-based landing guidance framework, while demonstrating its applicability through experiments.

B. Contribution

In line with our previous work [22], this paper adopts a more thorough technical approach, where robustness and reliability are improved. The main contributions are summarized as follows:

- A novel non-robocentric (offboard sensing and computing) autonomous landing framework is presented, and effective implementation is demonstrated.
- By taking landing safety corridors and dynamical constraints into consideration, a touchdown trajectory in the non-inertial frame is proposed to ensure the safety and smoothness of the landing process. Meanwhile, LED-based pose estimation method in the target local frame is adopted for feedback control.
- All above-described modules are integrated and and validated via a series of experiments with open-source hardware and software details[1] for future reference.

II. SYSTEM OVERVIEW

As mentioned, the proposed framework consists of two main modules, i.e., the landing guidance trajectory module and the relative pose estimation module, with the following submodules for:

- motion planning (Sec. II-A);
- finite state machine (Sec. II-B);
- marker detection (Sec. II-C);
- pose estimation (Sec. II-D).

Figure 2 illustrates the whole software architecture, in which the relationships between each module and its submodules is shown. Each module could be readily replaced by other hardware platforms and algorithms, making the proposed non-robocentric landing system a generalized framework. In this implementation, we exploit a relatively small scale system setup, in which we aim to land a 180 mm $\times$ 180 mm micro quadrotor onto a 430 mm $\times$ 430 mm landing pad installed on a ground vehicle.

A. Motion Planning

We formulate our guidance trajectory planner as a convex optimization problem and the objective is subject to visibility and dynamic constraints. Particularly, we represent the trajectory as a polynomial with Bernstein basis (Bézier curve) $b_n^i(t)$,

$$p(t) = \sum_{i=0}^{n} p_i t^i = \sum_{i=0}^{n} c^i b_n^i(t),$$
$$where\ \ b_n^i(t) = \binom{n}{i} t^i (1-t)^{n-i}, \tag{1}$$

where the trajectory is parameterized by t whereas c^i is the weighting coefficient of the basis, which could also be seen as the control points of the polynomial. Note that the polynomial is determined in the target local frame (P) of the landing platform. Bézier curve yields two useful properties for constraining the optimization problem, which are [23]:

1) *Convex Hull* Property. Based on the control points c^i, the generated trajectory $p_{\mathcal{B}}(t)$ is confined within the convex hull constructed by c^i's.
2) *Hodograph* Property. With the hodograph property, the derivatives of the Bézier curve are also Bézier curves, which could be expressed in terms of c^i.

Using the above characteristics, the trajectory could be constrained within a desired feasible region. To generate an optimal trajectory, we then minimize the integral of

[1]https://github.com/HKPolyU-UAV/alan.git

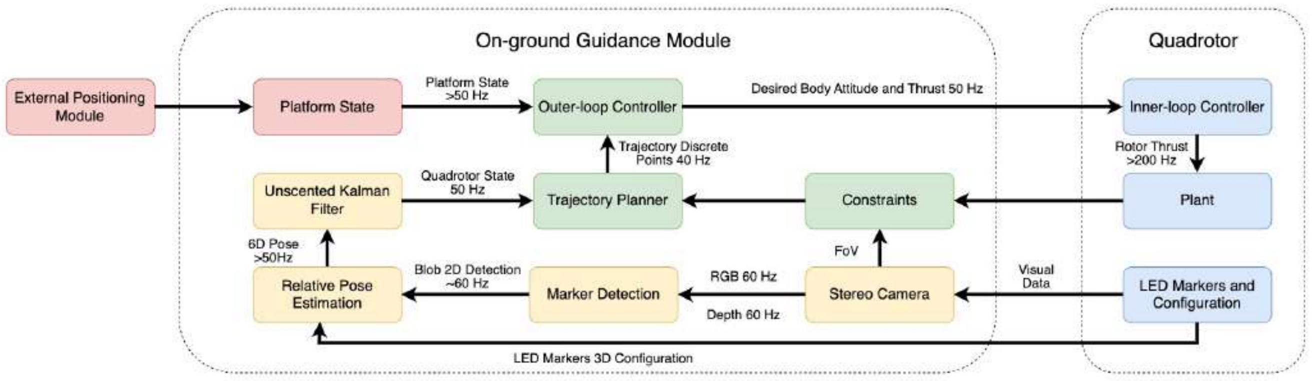

Fig. 2. The overall software architecture for the proposed implementation. The system could be discussed in two main parts, i.e., the on-ground guidance module and the airborne quadrotor. The on-ground module comprises two main submodules, for motion planning (colored in green) and relative pose estimation (colored in yellow). It also uses external positioning systems to localize itself. The quadrotor is equipped with an onboard controller and visible LED markers. Note that all modules mentioned above are replaceable, making it a generalized framework.

the squared 3^{rd}-derivative (i.e., jerk) of the trajectory with respect to the weights of the polynomial, i.e., the control points c^i. The objective cost function is shown below:

$$J = \int_{t_0}^{t} \left\| \frac{d^3 p_{\mathcal{B}}(t)}{dt^3} \right\|^2 dt = c^T M^T Q M c, \tag{2}$$

in which c is the optimization variables, M is the mapping matrix, and Q is the differential matrix.

As mentioned, in this implementation, we consider the visibility constraints and the dynamic constraints. Based on the effective perception range, we construct the visual safe flight corridor (VSFC), which is also expressed in target local frame P. In specifics, we first define a landing cubic free space ($\mathcal{F}$) based on the initial landing state (x_0) and the terminating state (x_T). Then, occupancies $\mathcal{O}$ within $\mathcal{F}$ is then excluded, where $\mathcal{F} \coloneqq \mathcal{F} \setminus \mathcal{O}$. Successively, $\mathcal{F}$ is segmented into two: *rear* space ($\mathcal{F}_R$) and *touchdown* space ($\mathcal{F}_T$), where we confine $\mathcal{F}_T$ with a touch down height (h_T) and exclude the space above h_T. $\mathcal{F}_T$ is deflated to ensure the final touch down segment of the trajectory has less vertical maneuver above the landing pad, so that the ground-effect is reduced. To further guarantee the continuity of the entire landing free space, $\mathcal{F}_R$ and $\mathcal{F}_T$ are enlarged with dilate coefficient ϵ_d to let $\mathcal{F}_R \cap \mathcal{F}_T \neq \emptyset$. The final set of landing free space is then denoted as $\mathcal{F}_L$. Lastly, the intersection between $\mathcal{F}_L$ and the FoV pyramid is returned. Also, to guarantee safety, the robot is modelled as a sphere with radius r_q. We then shrink $\mathcal{P}_{\mathcal{L}}$ with ϵ_s based on r_q, $\mathcal{P}_{\mathcal{L}} \coloneqq \epsilon_s \mathcal{P}_{\mathcal{L}}$. Note that VSFC is in fact a sequence of convex polyhedra, which could be therefore expressed as,

$$\mathcal{P}_{\mathcal{L}} = \{\sigma | a_i^T \sigma \leq b_i, \forall i = 0, 1, 2..., l\}. \tag{3}$$

Figure 3 shows an example of VSFC. Algorithm 1 summarizes the above procedure in psuedo-code form.

As for dynamic constraints, with the trajectory being planned within the target local frame, the velocity and acceleration limitations β_Q are transformed from the original body frame (Q) to target coordinate frame (P), as shown below,

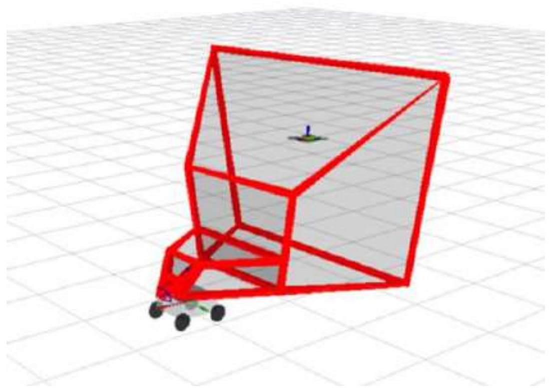

Fig. 3. An instance of the proposed VSFC. Herein, we utilize the visualization software package developed by Liu et al. [24].

$$\beta_D(t) = \mathcal{T}_Q^P \beta_Q - \beta_P(t), \mathcal{T}_Q^P \in SE(3). \tag{4}$$

In which, $\mathcal{T}_Q^P$ is the transformation matrix, while $\beta_P(t)$ stands for the velocity and acceleration of the reference frame at time t. On the other hand, equality constraints β_{eq} are derived based on the initial landing states (x_0), ending state (x_T), and continuity between each polynomial.

The final optimization problem is hence formulated as follows:

$$\begin{aligned} &\underset{c}{\text{minimize}} \;\; c^T M^T Q M c \\ &subject\ to \\ &\qquad \mathcal{A}_{eq} c = \beta_{eq} \\ &\qquad \mathcal{A}_d c \preceq \beta_{\mathcal{D}} \\ &\qquad \mathcal{A}_k c \preceq \beta_{\mathcal{P}}, \end{aligned} \tag{5}$$

Algorithm 1: VSFC Generation

```
1  end
2  Function VSFCGen($x_0$, $x_T$, $\mathcal{O}$, $h_T$, $\theta_w$, $\theta_w$):
3      if canLAND($x_0$, $x_T$) = false then
4          return $\langle fail, null \rangle$
5      $\mathcal{F} \leftarrow$ FreeGen($x_0$, $x_T$)
6      $\mathcal{F} :- \mathcal{F} \setminus \mathcal{O}$
7      $\langle \mathcal{F}_R, \mathcal{F}_T \rangle \leftarrow$ SplitSpace($F$)
8      $\mathcal{F}_T :-$ VerticalDeflate($\mathcal{F}_T$)
9      $\langle \mathcal{F}_R, \mathcal{F}_T \rangle :- \epsilon_d \langle \mathcal{F}_R, \mathcal{F}_T \rangle$ such that $\mathcal{F}_R \cap \mathcal{F}_T \neq \emptyset$
10     $\mathcal{P}_\mathcal{H} \leftarrow$ PyramidGen($\theta_w$, $\theta_w$)
11     $\langle \mathcal{P}_{\mathcal{L}R}, \mathcal{P}_{\mathcal{L}T} \rangle \leftarrow \langle \mathcal{F}_R \cap \mathcal{P}_\mathcal{H}, \mathcal{F}_T \cap \mathcal{P}_\mathcal{H} \rangle$
12     $\mathcal{P}_\mathcal{L} \leftarrow \langle \mathcal{P}_{\mathcal{L}R}, \mathcal{P}_{\mathcal{L}T} \rangle$
13     $\mathcal{P}_\mathcal{L} :- \epsilon_s \mathcal{P}_\mathcal{L}$
14     return $\langle success, \mathcal{P}_\mathcal{L} \rangle$
15 end
```

in which c is the optimization variable, and M and Q are the matrices derived in (2). $\mathcal{A}_{eq}$ denotes the affine relation matrix between control points c and equality constraints, while $\mathcal{A}_d$ and $\mathcal{A}_k$ map C to inequality constraints. β_{eq} and $\beta_\mathcal{D}$ are respectively the aforementioned equality constraint and inequality dynamic constraint, whereas $\beta_\mathcal{P}$ is the inequality constraint induced by (3). From (5), one could readily identify it as a convex problem, as it has a convex objective function and convex constraints. Furthermore, it is also a quadratic programming (QP) problem, which could be solved by off-the-shelf QP solvers.

B. Finite State Machine

To perform the entire landing mission, a finite state machine (FSM) is harnessed to control the overall process. The FSM comprises 5 states: searching, rendezvous, tracking, landing, and shutdown. Below describes them in detail.

1) *Searching*: To approach the vicinity of the landing platform, the quadrotor will plan a path from its current position to the rear area of the landing platforms using external positioning information (e.g. GNSS or motion capture).
2) *Rendezvous*: The sensor and computer on the landing platform will try to initialize the pose tracker at *Rendezvous* once the quadrotor is captured. After initialization, the stability of the pose estimation module will be evaluated; with a certain number of tracked epochs, the state will then be switched to *Tracking*.
3) *Tracking*: The quadrotor obtains direct control commands from the on-ground computer. To confirm the stability of both pose estimation and control commanding modules, this state makes the quadrotor follow the landing platform for a while. The mode is switched to *Landing* if it is deemed stable.
4) *Landing*: To begin with the *Landing* stage, the motion planning module optimizes a trajectory in the non-inertial frame. The software then guides the quadrotor based on the generated motion primitive.
5) *Shutdown*: The planner plans the guidance trajectory to approximate the center of the landing pad. The rotors of the quadrotor will be stopped once the relative distance is sufficiently small; the mission is then concluded.

Figure 4 shows the interrelations between each states.

C. Marker Detection

The markers are first attached directly to the quadrotor, where we set n as the number of markers and denote the marker configuration in quadrotor body frame (Q) as $\mathcal{M} = \{m_1, m_2, ...m_n\}$. Then, to detect the markers on the 2D image, conventional image processing methods are adopted, in which, we utilize the grayscale value and depth image information to retrieve possible pixels for blob extraction. Successively, blob detection is conducted to extract groups of pixels, in which the center coordinates of each blob group are inserted to detection candidates set $\mathcal{C}$.

In the presence of noises, solely relying on thresholding and blob detection is deemed insufficient. Therefore, to further reject outliers, at each detection, we also retrieve the raw 3D information of the blob centers within the camera frame with aligned depth image, which are denoted as set $PCD_\mathcal{C}$. With $PCD_\mathcal{C}$, the mean average deviation (MAD) of the 3D position $MAD(PCD_\mathcal{C})$ of each detection is calculated. The difference between the MAD values of blob detection and marker configuration is then calculated,

$$\Delta_{MAD} = MAD(PCD_\mathcal{C}) - MAD(\mathcal{M}). \tag{6}$$

If Δ_{MAD} exceeds a threshold λ_{MAD}, the program identifies the detection as corrupted and performs outlier rejection. Afterwards, outlier points are rejected by comparing each element in $PCD_\mathcal{C}$ with the transformed Euclidean center of $\mathcal{M}$, i.e., $\mathcal{M}' = \mathcal{T}_b^c \mathcal{M}$; with $\mathcal{T}_b^c$ being the estimated pose at time step $t = t_k - 1$. We then exclude the points ($\mathcal{C}'_O$) where its Euclidean distance exceeds threshold λ_d, $\mathcal{C} := \mathcal{C} \setminus \mathcal{C}'_O$. To further reduce noisy detection, K-means clustering [25] is applied to suppress the redundant detection ($\mathcal{C}'_R$) of each candidate, where $k = n$. The suppression then updates the set, $\mathcal{C} := \mathcal{C} \backslash \mathcal{C}'_R$. The final detection candidates $\mathcal{C}$ at each time step will then be passed to the pose estimation submodule (Sec. II-D). Algorithm 2 summarizes the method for marker blobs extraction.

Algorithm 2: Retrieval of Detection Candidates $\mathcal{C}$

```
1  Function Retrieve$\mathcal{C}$(GsImg, DpthImg):
2      Img :- Thres(GsImg, DpthImg)
3      $\mathcal{C} \leftarrow$ BlobExtract(Img)
4      $\mathcal{C}'_O \leftarrow$ BlobExtract($\mathcal{C}$)
5      $\mathcal{C} :- \mathcal{C} \setminus \mathcal{C}'_O$
6      $\mathcal{C} :-$ Kmeans($\mathcal{C}$)
7      return $\mathcal{C}$
8  end
```

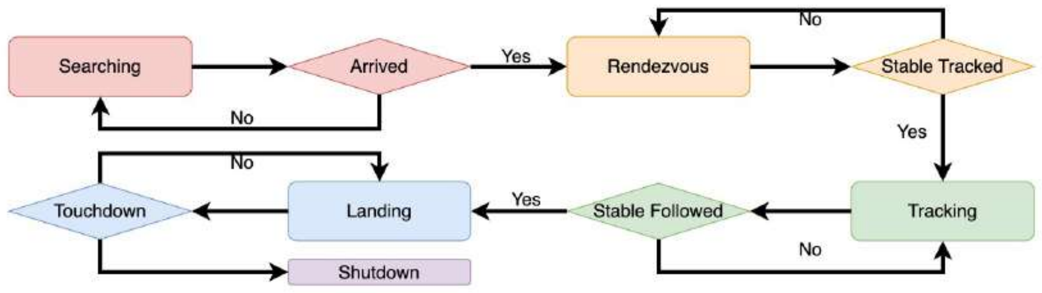

Fig. 4. The interrelations between each states.

D. Pose Estimation

To solve the 6D pose, the correct correspondence, i.e., the 2D-3D pair of $\mathcal{M}$ and $\mathcal{C}$, should be inferred. Herein, we adopt a similar method as presented in [26], yet with modifications. The process could be discussed respectively with two different stages: (1) initialization at $t = 0$ and (2) correspondence search at $t > 0$. At $t = 0$, a brute-force search method is used, where the pose of each correspondence permutation is calculated to acquire the correspondence with the lowest reprojection error. Meanwhile, as visible lights LED are utilized in this implementation, additional information from the RGB channels can reduce the search size: the HSV (Hue, Saturation, and Value) values of each LED blob could indicate wrong correspondences and exclude incorrect permutations during the inquiry. For each remaining permutation (m), a pose $\mathcal{T}_m$ is then solved with the EPnP algorithm [27]. The reprojection error (ε_m) is calculated with

$$\varepsilon_m = \sum_{i=1}^{n} \|K_c \mathcal{T}_m \mathcal{M}_i - \mathcal{C}_i\|_2{}^2, \forall m \in [1, 2, ..., m_0]. \qquad (7)$$

Note that K_c denotes the intrinsic matrix of the camera, and m_0 is the total number of permutations. $\mathcal{T}_m$, $\mathcal{M}_i$, and $\mathcal{C}_i$ are the aforementioned pose of each permutation, 3D position of each marker in body frame Q, and 2D information of each detection candidate, respectively. After the initial correspondence search ($t > 0$), we use the estimation at time t_{k-1} as priori estimation at t_k, which is denoted as $\hat{x}_k$. At each time step $t > 0$, reprojection of $\mathcal{M}$ onto 2D-image frame based on $\hat{x}_k$ is carried out, where reprojected points $\mathcal{M}_I$ are returned. Correspondences between the projected points $\mathcal{M}_I$ and the newly detected blobs $\mathcal{C}$ will then be searched, in which K-means clustering [25] is used again and $k = n$.

After securing the correct correspondence, we then utilize the result to retrieve the UAV pose. To yield a fast and accurate result, the EPnP algorithm [27] is adopted to solve the 6D pose information. To prevent PNP degeneration, colinear, coplanar, and symmetry geometries in $\mathcal{M}$ are avoided. After getting the initial estimate from EPnP, the result is then optimized through bundle adjustment within a single frame. The refinement result could be expressed as the following,

$$\mathcal{T}^{'} = \arg\min_{\mathcal{T}} \sum_{i=1}^{n} \|K_c \mathcal{T} \mathcal{M} - \mathcal{C}_i\|^2. \qquad (8)$$

By parametrization of exponential mapping, the above nonlinear optimization problem is then solved with the Gauss-Newton algorithm. The final estimated state is then forwarded to the motion planning module for feedback control.

III. RESULTS

A. Implementation Details

To validate the proposed framework, a quadrotor frame with a dimension of 180 mm $\times$ 180 mm is manufactured, while a Pixracer controller is installed with PX4 firmware [2]. A wireless communication module is also connected to the controller providing high bandwidth connectivity. 6 LED markers are also attached at the front of the quadrotor for the pose estimation.

Meanwhile, for the offboard modules, we use AgileX Scout Mini[3] ground vehicle as the main platform, where we place the landing pad (430 mm $\times$ 430 mm), a set of visual sensor (Intel RealSense D455), and a computer (Intel NUC minicomputer, NUC8i7BEH) upon it. The camera is mounted with an upward-facing angle of 20° and has a FoV of 87° $\times$ 58°. The computer runs several threads in a parallel fashion, including (1) pose estimation, (2) trajectory planner, (3) quadrotor outer-loop controller, and (4) ground vehicle path planner and controller. All modules above are programmed in C++/Python in the ROS framework. To solve the quadratic programming, we utilize OSQP [28], [29], a solver that provides efficient solutions within milliseconds and an easy interfacing module for integration.

On the other hand, experiments are done within a motion capture system-equipped arena, which it provides state estimation feedback for UGV and UAV at *Searching* and *Rendezvous* stage, acting as a "pseudo-GPS system". Figure 5 depicts the experimental setup.

B. Relative Pose Estimation

We compare the relative pose estimation to ArUco [30]. Moreover, to obtain the constellation of the LEDs as well as their positions in the body frame (Q) of the quadrotor,

[2]https://px4.io/
[3]https://global.agilex.ai/products/scout-mini

a motion capture system, i.e., VICON, is used to conduct calibration. Note that the motion capture system also offers sub-millimeter accurate ground truth reference. We then propose two experiments to evaluate the accuracy of pose estimation: (1) camera at still, and (2) camera in motion.

In the first experiment, the quadrotor is preprogrammed to conduct a random maneuver in front of the camera. The distance between the quadrotor and the ground vehicle varies from 0.4 m to 3 m. In the second experiment, to emulate a landing scenario, we test the relative pose estimation module with the camera moving dynamically. In which, the ground vehicle moves randomly, while the quadrotor is programmed to follow the landing platform. Table I shows the performance.

TABLE I
COMPARISON BETWEEN DIFFERENT POSE ESTIMATION METHODS

	Static		Dynamic		Unit
	LED	ArUco	LED	ArUco	
Mean Error (Position)	0.045	0.061	0.055	0.121	m
Std. Dev. (Position)	0.031	0.094	0.083	0.035	m
Mean Error (Orientation)	0.003	0.210	0.021	0.036	rad
Std. Dev. (Orientation)	0.052	0.481	0.042	0.729	rad
Detection Success Rate	0.992	0.491	0.999	0.290	N/A

From above, it could be seen that, with the same setup, the proposed implementation yields a better result in terms of translation and orientation error, standard deviation of error, and success rate. Note that the success rate is equivalent to the percentage of the frames with reprojection error lower than 10 pixels; the metric for orientation comparison is expressed in angle-axis representation.

C. Landing Experiments

We then present landing experiments, in which the UGV follows a circular path and tries to guide the quadrotor if found and tracked. The maximum velocity of the camera is estimated to be 2 m/s, which is faster than the experiments reported in [10] with a straight maneuver of the ground vehicle.

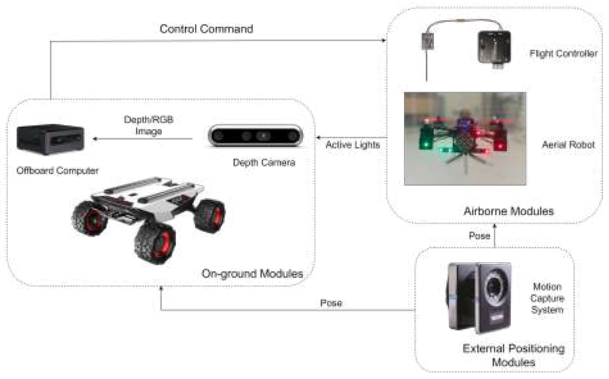

Fig. 5. Experimental setup for the proposed implementation.

The final experiment results are shown in Figure 6, in which the position and velocity of both UAV and UGV are presented. Note that the data are recorded after the system enters the *Rendezvous* state. In the absence of airborne sensors and computers, it could be seen that the on-ground software could guide the quadrotor to perform landing smoothly. In addition, by setting the dynamic constraints in relative frame (as mentioned in (4)), the generated trajectory is in compliance with the dynamic limitations. We have conducted the experiments with different ground vehicle velocity settings, yet for brevity, only the data with the highest velocity are shown here. Furthermore, figure 7 further shows the relative pose estimation error, while figure 8 illustrates the trajectory tracking error. The tracking error is defined as the difference between the setpoint position and actual state.

It could be seen that both pose estimation error and trajectory tracking error have a relatively large value during the final landing stage; nevertheless, under such ground vehicle velocity settings and wireless communication setup, the error is deemed to be within an acceptable margin. Therefore, from all above, the landing objective is considered achieved: despite the absence of exteroceptive sensors and computers on the quadrotor, a smooth landing performance was displayed. This implies the plausibility of a non-robocentric dynamic landing framework. Lastly, a video is attached for clearer visualization for readers.

IV. CONCLUSIONS

In this paper, an unconventional dynamic landing configuration for quadrotor is proposed, in which onboard sensors and computers on UAVs are unequipped for landing purpose, making it a "non-robocentric" framework. Furthermore, to validate such a framework, we propose an effective implementation, where the relative pose estimation and motion planner are included, and a series of experiments is conducted.

From the implementation results, we consider the framework an applicable option for dynamic landing, in which, with a sub-meter tracking error, the non-robocentric framework achieves similar results as conventional methods. Therefore, it is concluded that with robust relative pose estimation and a well-designed guidance method, the proposed framework is a viable option to deal with the dynamic landing problem. And in particular, such a methodology could be specifically adopted when equipping heavy and power-consuming sensors and computers is not a necessity.

In the near future, we plan to improve the relative pose estimation for such a landing framework through multi-modal sensor fusion. Meanwhile, we also propose to conduct experiments on outdoor moving vehicles and marine vessels to test the real-world practicality.

REFERENCES

[1] P. Tokekar, J. Vander Hook, D. Mulla, and V. Isler, "Sensor planning for a symbiotic uav and ugv system for precision agriculture," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1498–1511, 2016.
[2] H. Chen and P. Lu, "Real-time identification and avoidance of simultaneous static and dynamic obstacles on point cloud for uavs navigation," *Robotics and Autonomous Systems*, vol. 154, p. 104124, 2022.

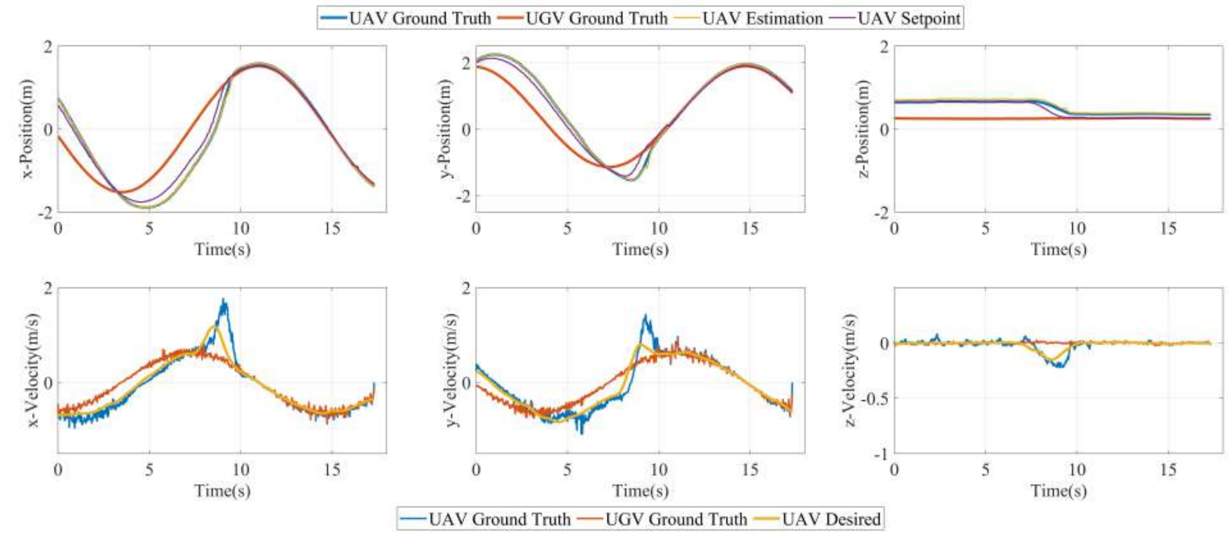

Fig. 6. The position and velocity information of the landing mission. In which, the blue curve indicates the quadrotor ground truth, the orange curve shows the ground vehicle ground truth, the yellow curve represents the quadrotor estimation from ground, and the purple curve denotes the setpoint for the aerial robot.

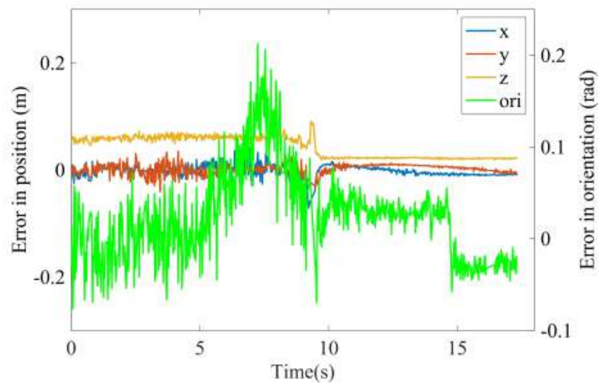

Fig. 7. The error of the relative pose estimation during landing.

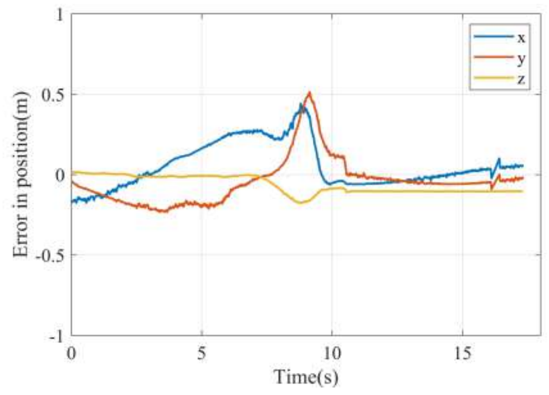

Fig. 8. The tracking error of guidance trajectory during landing. Similar to Figure 7, the largest error occurred when approaching the landing pad due to aggressive maneuver. Yet, the error is deemed within acceptable region.

[3] L.-Y. Lo, C. H. Yiu, Y. Tang, A.-S. Yang, B. Li, and C.-Y. Wen, "Dynamic object tracking on autonomous uav system for surveillance applications," *Sensors*, vol. 21, no. 23, p. 7888, 2021.

[4] J. Sun, B. Li, Y. Jiang, and C.-Y. Wen, "A camera-based target detection and positioning uav system for search and rescue (sar) purposes," *Sensors*, vol. 16, no. 11, p. 1778, 2016.

[5] S. Chen, Y. Feng, C.-Y. Wen, Y. Zou, and W. Chen, "Stereo visual inertial pose estimation based on feedforward and feedbacks," *IEEE/ASME Transactions on Mechatronics*, 2023.

[6] Q. Zhou, L.-Y. Lo, B. Jiang, C.-W. Chang, C.-Y. Wen, C.-K. Chen, and W. Zhou, "Development of fixed-wing uav 3d coverage paths for urban air quality profiling," *Sensors*, vol. 22, no. 10, p. 3630, 2022.

[7] H. Huang, A. V. Savkin, and C. Huang, "Control of a novel parcel delivery system consisting of a uav and a public train," in *2019 IEEE 17th International Conference on Industrial Informatics (INDIN)*, vol. 1. IEEE, 2019, pp. 1047–1050.

[8] H. Menouar, I. Guvenc, K. Akkaya, A. S. Uluagac, A. Kadri, and A. Tuncer, "Uav-enabled intelligent transportation systems for the smart city: Applications and challenges," *IEEE Communications Magazine*, vol. 55, no. 3, pp. 22–28, 2017.

[9] D. Lee, T. Ryan, and 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, pp. 971–976.

[10] D. Falanga, A. Zanchettin, A. Simovic, J. Delmerico, and 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, pp. 200–207.

[11] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "Svo: Semidirect visual odometry for monocular and multicamera systems," *IEEE Transactions on Robotics*, vol. 33, no. 2, pp. 249–265, 2016.

[12] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient motion primitive for quadrocopter trajectory generation," *IEEE transactions on robotics*, vol. 31, no. 6, pp. 1294–1310, 2015.

[13] S. Yang, S. A. Scherer, and A. Zell, "An onboard monocular vision system for autonomous takeoff, hovering and landing of a micro aerial vehicle," *Journal of Intelligent & Robotic Systems*, vol. 69, no. 1, pp. 499–515, 2013.

[14] P. Vlantis, P. Marantos, C. P. Bechlioulis, and K. J. Kyriakopoulos, "Quadrotor landing on an inclined platform of a moving ground vehicle," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 2202–2207.

[15] Y. Meng, W. Wang, H. Han, and J. Ban, "A visual/inertial integrated landing guidance method for uav landing on the ship," *Aerospace Science and Technology*, vol. 85, pp. 474–480, 2019.

[16] Y. Qi, J. Jiang, J. Wu, J. Wang, C. Wang, and J. Shan, "Autonomous landing solution of low-cost quadrotor on a moving platform,"

Robotics and Autonomous Systems, vol. 119, pp. 64–76, 2019.

[17] A. Paris, B. T. Lopez, and J. P. How, “Dynamic landing of an autonomous quadrotor on a moving platform in turbulent wind conditions,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 9577–9583.

[18] C. Martínez, P. Campoy, I. Mondragón, and M. A. Olivares-Méndez, “Trinocular ground system to control uavs,” in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Ieee, 2009, pp. 3361–3367.

[19] W. Kong, D. Zhang, X. Wang, Z. Xian, and J. Zhang, “Autonomous landing of an uav with a ground-based actuated infrared stereo vision system,” in *2013 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2013, pp. 2963–2970.

[20] W. Kong, D. Zhou, Y. Zhang, D. Zhang, X. Wang, B. Zhao, C. Yan, L. Shen, and J. Zhang, “A ground-based optical system for autonomous landing of a fixed wing uav,” in *2014 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2014, pp. 4797–4804.

[21] T. Ferreira, A. Bernardino, and B. Damas, “6d uav pose estimation for ship landing guidance,” in *OCEANS 2021: San Diego–Porto*. IEEE, 2021, pp. 1–10.

[22] C.-W. Chang, L.-Y. Lo, H. C. Cheung, Y. Feng, A.-S. Yang, C.-Y. Wen, and W. Zhou, “Proactive guidance for accurate uav landing on a dynamic platform: A visual–inertial approach,” *Sensors*, vol. 22, no. 1, p. 404, 2022.

[23] R. F. Riesenfeld, *Applications of b-spline approximation to geometric problems of computer-aided design.* Syracuse University, 1973.

[24] S. Liu, M. Watterson, K. Mohta, K. Sun, S. Bhattacharya, C. J. Taylor, and V. Kumar, “Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-d complex environments,” *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1688–1695, 2017.

[25] J. A. Hartigan and M. A. Wong, “Algorithm as 136: A k-means clustering algorithm,” *Journal of the royal statistical society. series c (applied statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[26] M. Faessler, E. Mueggler, K. Schwabe, and D. Scaramuzza, “A monocular pose estimation system based on infrared leds,” in *2014 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2014, pp. 907–913.

[27] V. Lepetit, F. Moreno-Noguer, and P. Fua, “Ep n p: An accurate o (n) solution to the p n p problem,” *International journal of computer vision*, vol. 81, pp. 155–166, 2009.

[28] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, “OSQP: an operator splitting solver for quadratic programs,” *Mathematical Programming Computation*, vol. 12, no. 4, pp. 637–672, 2020. [Online]. Available: https://doi.org/10.1007/s12532-020-00179-2

[29] G. Banjac, B. Stellato, N. Moehle, P. Goulart, A. Bemporad, and S. Boyd, “Embedded code generation using the OSQP solver,” in *IEEE Conference on Decision and Control (CDC)*, 2017. [Online]. Available: https://doi.org/10.1109/CDC.2017.8263928

[30] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, “Automatic generation and detection of highly reliable fiducial markers under occlusion,” *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014.