Journal of Guidance, Control, and Dynamics, Availbale online 19th January 2026
DOI: 10.2514/1.G009512

Visual Servoing MPC for Autonomous Shipboard Rotorcraft Landing in High Sea States

Luís G. L. de Paula*, Jose M. Castiblanco Quintero†, Hyo-Sang Shin‡, and Antonios Tsourdos§
Cranfield University, Faculty of Engineering and Applied Sciences, MK43 0AL, Cranfield, UK

Autonomous rotorcraft landing on ships under harsh sea state conditions is a challenging task. Wave-induced oscillations of the landing deck require precisely timed touchdowns to prevent unsafe conditions such as rollover. Moreover, a key challenge is to enable autonomous landing without relying on complex hardware installed on the deck. This paper presents a novel approach that integrates visual servoing with model predictive control to overcome the requirement of hardware on deck, while introducing vessel motion into the landing decision using cost barrier functions. Short-term online forecast of landing pad states is achieved using a combination of Fast Fourier Transform and Kalman Filter. The proposed model was evaluated through scaled-down simulations and indoor flight tests replicating harsh sea conditions. Two alternatives were compared to the baseline image-based visual servoing: a forecast-enhanced IBVS and the proposed MPC. Simulations show that the baseline visual servoing often fails to meet roll constraints at touchdown since it relies on current vessel states. The proposed controller outperforms both the baseline and its forecast-enhanced version, achieving an improved landing success rate under Sea State 6 with significant roll oscillations. Success criteria included roll constraints and touchdown accuracy, supported by statistical database with multiple landing trials.

I. Introduction

Unmanned Aerial Vehicles (UAVs) have been increasingly deployed in offshore environments. In these scenarios, vertical take-off and landing (VTOL) rotorcraft have proven to be valuable assets to support tasks such as infrastructure inspection, search and rescue (SAR) missions, and surveillance and reconnaissance operations. However, extending the operational flight envelope for these critical missions requires overcoming harsh maritime conditions. These harsh conditions can occur up to 34% of the time as per the Open Ocean Waters database [1]. A key challenge is

*PhD Student, Cranfield University, Faculty of Engineering and Applied Sciences, MK43 0AL, Cranfield, UK, luis.leandrodepaula@cranfield.ac.uk (corresponding Author)
†Research Fellow, Cranfield University, Faculty of Engineering and Applied Sciences, MK43 0AL, Cranfield, UK, jocasqui@upv.edu.es
‡Professor, Cranfield University, Faculty of Engineering and Applied Sciences, MK43 0AL, Cranfield, UK and Cho Chun Shik Graduate School of Mobility, KAIST, Daejeon, 34141, South Korea, hyosangshin@kaist.ac.kr (corresponding Author)
§Professor, Faculty of Engineering and Applied Sciences, MK43 0AL, Cranfield, UK, a.tsourdos@cranfield.ac.uk
This paper is an extended version of AIAA-2025-3547, "Visual Servoing MPC Framework for Shipboard Autonomous Landing", presented at the AIAA Aviation Forum and Ascend, Las Vegas, NV, July 2025

Published by AIAA. This is the Author Accepted Manuscript issued with: Creative Commons Attribution License (CC:BY 4.0).
The final published version (version of record) is available online at DOI:10.2514/1.G009512. Please refer to any applicable publisher terms of use.

to design UAV controllers to perform autonomous landings on vessels subjected to harsh sea motion, particularly in sea states (SS) higher than 5 based on the Beaufort scale [2]. The UAV must touch down only when the roll motion of the deck falls within an often restrictive landing envelope designed to prevent rollover and landing gear damage. In recent literature, autonomous flight tests based on traditional image based visual servoing (IBVS) controllers provided good touchdown precision under mild sea states [3–5]. The IBVS underlying strategy relies on monitoring the current states of the vessel to trigger discrete hover transitions until touchdown. However, this IBVS controller yields unsafe conditions as it is not capable of meeting the roll limits when the landing pad is under harsh sea motion. More recent approach introduced nonlinear model predictive controllers (NMPC) based on visual servoing kinematics [6, 7]. These control solutions provided accurate tracking while handling field-of-view constraints. Nevertheless, they do not address the landing phase or assume a static target. Therefore, this paper introduces a short-term forecast within the visual servoing predictive framework to cover the autonomous landing procedure on a platform under harsh sea state motion.

The scaled-down flight tests in recent literature provide valuable insight into the feasibility of proposed autonomous controllers. In [8], a novel vision-based approach used a gyro-stabilized horizon bar as a tracking reference. Flight tests were performed with a small-scale UAV. The relative pose was estimated using classical PnP (Perspective-n-Point) with a Kalman filter. A close-range PID controller, enhanced with a probabilistic filter on the derivative term to handle pose outliers, enabled precise landings within a 0.35 m × 0.35 m area on a forward moving target under wind disturbances. Although the research covered landing pad oscillations equivalent to SS 6, landing decisions were based only on a spatial threshold above the deck. Feed forward IBVS (FF-IBVS) using virtual image planes (VIP) has also shown successful landing rates in moderate sea states. In [3], outdoor flight tests with a 5 kg payload UAV demonstrated an average touchdown error of 0.2 m against a moving target with 6 m/s forward speed and 6 DoF oscillations simulating SS 4. In [4], a touchdown error of 5.5 cm was achieved with outdoor tests using a small scale UAV and a forward moving platform under manual roll and pitch oscillations. Although precise landings were achieved in [3, 4], the landing criteria of the IBVS controllers were based solely on the current states of the ship, which poses a challenge when extended to harsh sea conditions due to the fast dynamics of the landing deck. In [5], authors proposed a finite-time dynamic visual service (FTDVS), which uses optical flow during the landing phase while estimating the period of the deck oscillations. However, flight test validation was limited to targets with simplified sinusoidal oscillations. To further include target state forecast within the landing criteria, Model Predictive Control (MPC) were flight tested under harsh sea state [9, 10]. In [10], the authors presented a novel scaled-down QP guidance with a prediction horizon (PH) update scheme for autonomous deck landing. The target states were estimated using an Auto-Regressive (AR) model based on pose from an Optical Motion System (OMS). The research focused on direct landing instead of the traditional vertical landing procedure. In [9], a novel path planning MPC with a nonlinear estimator used 0.5 s state forecasts to improved vertical landing performance. The cost function included barrier terms based on ship states and UAV height. The forecast algorithm used an FFT model with a Discrete Kalman Filter (FFT-DKF). Under harsh SS, simulations achieved 94%

landings within 10° of target roll, outperforming the SH-NMPC developed in [11]. An enhanced version of the same algorithm was applied later in [12] for visual servoing during hover tracking. Rather than using simplified sinusoidal oscillation, target oscillations were simulated using a Response Amplitude Operators (RAO) similar to [13]. In [7, 14], an NMPC framework was proposed that includes both UAV and visual servoing dynamics to address visibility and input constraints. The approach demonstrated good tracking performance during flight tests against static targets. In [6], the authors introduced an online MPC-IBVS based on image moments that covers the attitude control loop to handle long-term FoV constraints. Although the controller achieves robust performance under bounded external disturbances, the study did not included the target states within the landing criteria. Overall, visual servoing approaches demonstrated good landing precision, but were either tested only under mild sea state conditions or failed to account for the target attitude within the landing criteria.

This work presents a visual servoing non-linear model predictive control (VS-NMPC) for autonomous UAV landing on a vessel under harsh sea state (SS 6). Previous literature based on FF-IBVS [3, 4] either ignores attitude constraints or relies on current target states to trigger landing. This work demonstrates that these approaches lead to terminal constraint violations in high sea states, increasing rollover risk. Moreover, recent predictive visual servoing controller [5, 6] are restricted to simplified sinusoidal oscillations or do not incorporate target states within landing criteria. The proposed controller incorporates barrier cost functions based on target states to meet roll angle constraints during touchdown. Realistic target oscillations are achieved by using Response Amplitude Operators (RAO) from a large scale destroyer. The framework features an online deck attitude forecasting module using FFT-DKF that relies on vision-based pose estimation. As a result, minimal communication with the landing platform is required, since only the forward speed of the ship is streamed to the UAV to be used as a feed-forward term. A cross-based validation with simulations and indoor flight tests using a small-scale UAV shows that a higher rate of safe landings is achieved compared to existing IBVS controllers [3, 4].

This paper is organized as follows. Section II introduces the vessel motion model derived from RAO, as well as the baseline feedforward IBVS controller using image moment features that will be used within the VS-NMPC formulation. Section III describes the proposed framework including details of the target detection pipeline, the target state forecast using FFT-DKF, and the design of the VS-NMPC. Section IV details the hardware setup, the scaled-down landing procedure, and the scenarios tested in simulation and flight tests. Section V shows the performance comparison between the baseline IBVS controller and the VS-NMPC. First, simulation results highlight the improvement of safe landing rates using the scale-down landing profile. Next, indoor flight tests provide a proof of concept under real-time processing conditions. The advantages of VS-NMPC over an enhanced version of the baseline IBVS are discussed taking into account both roll constraints and landing precision. Finally, the concluding remarks and future work are summarized in Section VI.

II. Modeling: Target motion and IBVS formulation

A. Coordinate frames and notations

Frame definitions are given in Fig. 1. Fig. 1(a) details the UAV body frame (S_b), Inertial frame (S_w), and Landing pad (or target) frame (S_t). P and R are the position and Euler angles of the UAV from the world frame, while P_t and R_t are for the target. The camera is assumed to be mounted in a fixed, downward-facing position. A virtual camera frame that compensates for UAV attitude is introduced to prevent incorrect velocity inputs [3]. Given the camera position O_c, both original camera frame (S_c) and virtual camera frame (S_v) are defined in Fig. 1(b). p_i^c and p_i^v are the position of the i-th detected feature in the original and virtual camera frames, respectively. The camera is assumed to be fixed in the downward direction (z_b) with a focal length of $f = ||\boldsymbol{f}'||$, where $\boldsymbol{f}' = [f_u f_v]$. Fig. 1(c) shows the projection of this feature into the original (I_c') and virtual image frames (I_v'). When centered in the image principal point these frames are defined as I_c and I_v. $\boldsymbol{V}_c^c = \left[V_x^c, V_y^c, V_z^c\right]^T$ and $\boldsymbol{\omega}_c^c = \left[\omega_x^c, \omega_y^c, \omega_z^c\right]^T$ are the camera velocity in the original camera frame, while $\boldsymbol{V}_c^v = \left[V_x^v, V_y^v, V_z^v\right]^T$ and $\boldsymbol{\omega}_c^v = \left[\omega_x^v, \omega_y^v, \omega_z^v\right]^T$ are in the virtual camera frame. The sigmoid function is defined as $\sigma(x) = (1 + e^{-x})^{-1}$, a n-th square identity matrix, as $\mathbb{I}_n$, and $\odot$ is the Hadamard product.

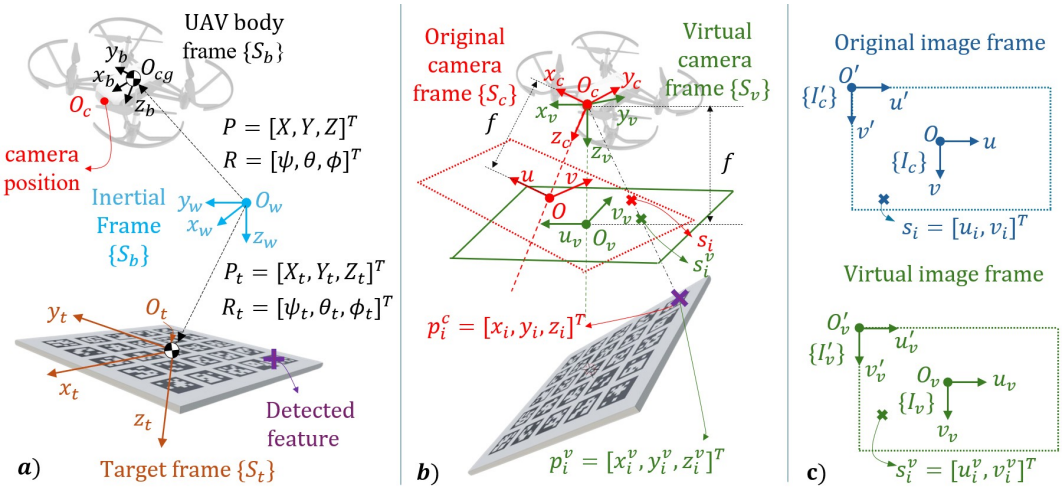

Fig. 1 Coordinate frame definitions: a) UAV and target frames, b) UAV camera frames, and c) camera image plane frames.

B. Vessel Motion Model

The motion of the vessel in the harsh sea state is simulated using Response Amplitude Operators (RAO). Fundamentally, RAO are transfer functions that output the motion of each state, $\eta \in [X_t, Y_t, Z_t, \psi_t, \theta_t, \phi_t]$, based on wave elevation inputs. The RAO are a function of the forward speed of the vessel (U), wave encounter angle (χ), and wave encounter frequency (ω_e); hence, $RAO_\eta(U, \chi, \omega_e)$. Here, it is assumed that the RAO is tabulated for a discrete number of $\omega_{e,k}$. A Bretschneider spectra can be used to model the power density of wave elevation inputs. Assuming fully developed long-crested seas, this is given by Eq. 1.

$$S(\omega) = \frac{5}{16}\frac{\omega_m^4}{\omega^5}H_{1/3}^2 e^{-5\omega_m^4/4\omega^4} \tag{1}$$

where ω are the wave input frequencies, ω_m is the modal wave frequency, and $H_{1/3}$ is the significant wave height. Both ω_m and $H_{1/3}$ can be found as a function of the sea state code (SS). Next, the effects of wave encounter angle and Doppler effect due to ship forward speed are introduced using a spectra transformation given by Eq. 2.

$$S_{\chi,U}\left(\omega_e\right) = \frac{S(\omega)}{\left|1 - \frac{2\omega U}{g}\cos\chi\right|}, \quad \omega_e = \omega - \frac{\omega^2 U\cos\chi}{g} \tag{2}$$

Based on the RAO definition, the spectra distribution for each state is given by Eq. 3.

$$S_\eta(\omega_e) = |RAO_\eta(U,\chi,\omega_e)|^2 S_{\chi,U}\left(\omega_e\right) \tag{3}$$

Finally, the vessel motion can be modeled as a collection of sinusoids with random phases as in Eq. 4.

$$\eta(t) = \sum_{k=1}^{N_\omega} \bar{\eta}_k \cos\left(\omega_{e,k}t + \gamma_k\right) \tag{4}$$

where N_ω is the total number of sinusoids, γ_k are the random wave phases. Here, the discrete $\omega_{e,k}$ from the tabulated RAO range in $k \in [0, N_\omega + 1]$. The amplitude $\bar{\eta}_k$ is given by Eq. 5. Note that, from Eq. 4, $\eta(t)$ can be converted to a specific point of interest, such as the center of the deck.

$$\bar{\eta}_k = \sqrt{2\int_{\omega_{e,k}-0.5(\omega_{e,k}-\omega_{e,k-1})}^{\omega_{e,k}-0.5(\omega_{e,k+1}-\omega_{e,k})} S_{\chi,U}\left(\omega_e\right) d\omega_e} \tag{5}$$

C. Baseline Visual Servoing Controller

The IBVS maps the time-varying motion of visual features onto a two-dimensional image plane. Although depth estimation is required, the controller can be formulated without relative pose. Assuming a pinhole camera, the image plane kinematics is given by

$$\dot{\boldsymbol{s}}_{\boldsymbol{i}} = \hat{\boldsymbol{J}}_p\left(\boldsymbol{s}_i, \hat{z}_i, \boldsymbol{f'}\right)\boldsymbol{v_c} \tag{6}$$

$$\hat{\boldsymbol{J}}_p\left(\boldsymbol{s}_i, \hat{z}_i, \boldsymbol{f'}\right) = \begin{bmatrix} -\frac{f_u}{\hat{z}_i} & 0 & \frac{u_i}{\hat{z}_i} & \frac{u_i v_i}{f_v} & -f_u - \frac{u_i^2}{f_u} & v_i\frac{f_u}{f_v} \\ 0 & -\frac{f_v}{\hat{z}_i} & \frac{v_i}{\hat{z}_i} & f_v + \frac{v_i^2}{f_v} & -\frac{u_i v_i}{f_u} & -u_i\frac{f_v}{f_u} \end{bmatrix} \tag{7}$$

where $\boldsymbol{\nu}_c = \left[\boldsymbol{V}_c^c, \boldsymbol{\omega}_c^c\right]^T$ is the camera velocity in S_c, $\hat{z}_i$ the depth estimate in S_c, and $\hat{\boldsymbol{J}}_p$ the image Jacobian matrix.

Hence, the IBVS is based on a 6 DoF velocity control ($\boldsymbol{\nu}_c \in \mathbb{R}^6$) using the pseudo-inverse of Eq. 6. However, this is not achievable for the quadrotor since rate, attitude and speed controllers are coupled. Moreover, the IBVS could trigger inputs in opposite directions [3]. A VIP provides a transformation to compensate for UAV attitude when using a fixed downward-facing camera [15]. The transformation from S_c to S_v is given by

$$\boldsymbol{p}_i^v = R_{x_c'}(-\theta) R_{y_c}(\phi) \boldsymbol{p}_i^c, \tag{8}$$

where x_c' is an intermediate axis. Given a virtual camera with the same intrinsic parameters, the VIP coordinates are defined as

$$\boldsymbol{s}_i^v = \frac{f}{z_i^v} \begin{bmatrix} x_i^v \\ y_i^v \end{bmatrix} = \begin{bmatrix} \frac{(u_i \cos\phi - f_u \sin\phi) f_u f_v}{u_i f_v \cos\theta \sin\phi + v_i f_u \sin\theta + f_u f_v \cos\phi \cos\theta} \\ \frac{(-u_i f_v \sin\phi \sin\theta + v_i f_u \cos\theta - f_u f_v \sin\theta \cos\phi) f_v}{u_i f_v \cos\theta \sin\phi + v_i f_u \sin\theta + f_u f_v \cos\phi \cos\theta} \end{bmatrix} \tag{9}$$

As a result, there are no changes in $\boldsymbol{s}_i^v$ due to $\omega_{x,c}$ or $\omega_{y,c}$. Finally, visual servoing dynamics can be simplified as

$$\dot{\boldsymbol{s}}_i^v = \begin{bmatrix} -\frac{f_u}{\hat{z}_i^v} & 0 & \frac{u_i^v}{\hat{z}_i^v} & v_i^v \frac{f_u}{f_v} \\ 0 & -\frac{f_v}{\hat{z}_i^v} & \frac{v_i^v}{\hat{z}_i^v} & -u_i^v \frac{f_v}{f_u} \end{bmatrix} \begin{bmatrix} V_x^v \\ V_y^v \\ V_z^v \\ \omega_z^v \end{bmatrix} \tag{10}$$

One downside of using feature points is that they lead to slower altitude convergence [6]. They also don't scale well in the MPC framework, since the number of states becomes $2n$. To simplify the visual servoing kinematics, image moments can be used instead [16]. In this work, the tracked features are the four corners of a square. Using the definitions $m_{ij} = \sum_{k=1}^{n} u_k^i v_k^j$, $u_g = m_{10}/m_{00}$, and $v_g = m_{01}/m_{00}$, the centered moments are given in Eq. 11, while the image moments are shown in Eqs. 12 and 13.

$$\mu_{ij} = \sum_{k=1}^{n} \left(u_k^v - u_g^v\right)^i \left(v_k^v - v_g^v\right)^j \tag{11}$$

$$q_h = \sqrt{\frac{a^*}{a}}, \quad q_u = \frac{u_g^v}{f} q_h, \quad q_v = \frac{v_g^v}{f} q_h \tag{12}$$

$$q_\psi = atan2\left(v_2^v + v_3^v - 2v_g^v, u_2^v + u_3^v - 2u_g^v\right) \tag{13}$$

where $a = \mu_{20} + \mu_{02}$, $a^\star$ is the value a at the desired relative height $z_i^{\star v}$ (or $z^\star$ for simplicity), and q_ψ is defined

as in [6] to uniquely define yaw of a coplanar square with corners (u_1^v, v_1^v), (u_2^v, v_2^v), (u_3^v, v_3^v), and (u_4^v, v_4^v). Following [16], if there is no heave motion, it can be shown that $z\sqrt{a} = z^\star\sqrt{a^\star}$. From Eqs. 11, 12, and 13, the servoing dynamics can be simplified as follows:

$$\dot{\boldsymbol{q}} = \boldsymbol{J}_q(\boldsymbol{q}, z^\star)\boldsymbol{V}_r^v \tag{14}$$

where

$$\boldsymbol{J}_q = \begin{bmatrix} -\frac{1}{z^\star} & 0 & 0 & q_v \\ 0 & -\frac{1}{z^\star} & 0 & -q_u \\ 0 & 0 & -\frac{1}{z^\star} & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \tag{15}$$

, $\boldsymbol{q} = [q_u, q_v, q_h, q_\psi]^T$, and $\boldsymbol{V}_r^v = [v_x^v, v_y^v, v_z^v, \omega_z^v]^T$. In particular, this formulation decouples its state dimension from the number of reference feature points, offering a more scalable representation. Moreover, it does not require depth estimation as it is implicit within the image moments. It should be noted that $n \geq 4$ feature points are required. Finally, conventional IBVS introduces the proportional controller $\dot{\boldsymbol{q}} = \boldsymbol{\lambda}(\boldsymbol{q}^\star - \boldsymbol{q})$, where $\boldsymbol{\lambda} = diag(\lambda_u, \lambda_v, \lambda_h, \lambda_\psi)$, and $\boldsymbol{q}^\star$ are the reference image moments. Therefore, the control input can be written as $\boldsymbol{V}_r^v = \boldsymbol{\lambda}\boldsymbol{J}_q^{-1}(\boldsymbol{q}, z^\star)(\boldsymbol{q}^\star - \boldsymbol{q})$.

III. Proposed framework

A schematic overview of the proposed framework is summarized in Fig. 2. The main contributions related to the landing pad detection pipeline, the vessel state forecast, and the VS-NMPC are highlighted within the overall architecture. Therefore, the following subsections describe the main contributions for each of these stages. As an initial proof of concept, it is assumed that the moving target has only roll oscillations.

A. Landing Pad Detection

The markers are detected using sub-pixel accuracy for corner positions. The reference features are the center point of each marker. In this paper, two detection layers were added. A square detection is performed to enhance robustness against partial occlusions mainly due to communication packet losses. Following this, a quadrilateral interpolation method based on [3] is applied to mitigate the tracking oscillations induced by the roll motion of the target.

1. Detection of landing pad corners

The corners of a reference square formed by the centers of the AruCo markers are used as tracking features. A binary detection matrix $\mathbf{D} \in \mathbb{Z}^{n\times n}$ is defined such that $D_{ij} = 1$, if the marker is detected on the position of the (i, j)

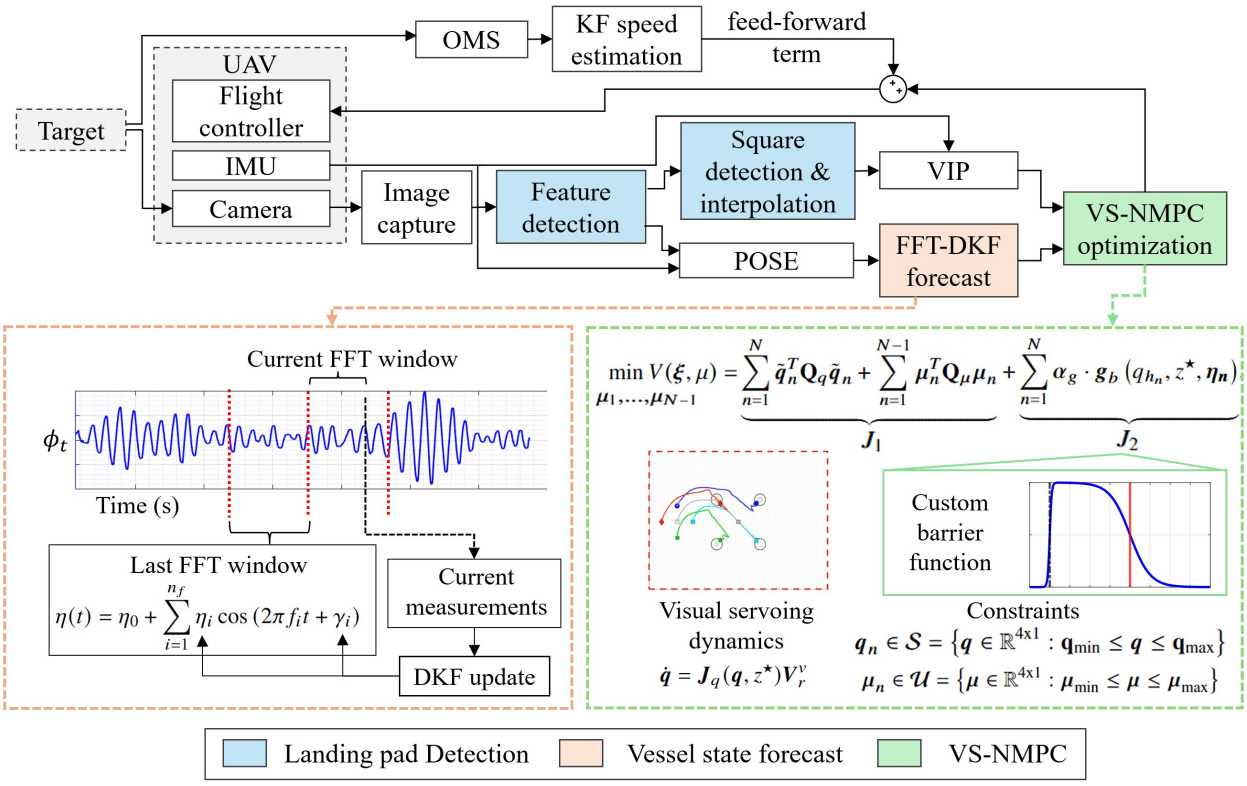

Fig. 2 Overview of the proposed VS-NMPC framework.

board, and 0 otherwise. Given $k \in [1, n-1]$, an existence check is done to all $k \times k$ sub-matrices, this results in a square detection mask $\mathbf{M}^{(k)} \in \mathbb{Z}^{(n-k)\times(n-k)}$ with elements defined as:

$$M_{ij}^{(k)} = \begin{cases} 1 & \text{if } D_{ij} = D_{i+k,j} = D_{i,j+k} = D_{i+k,j+k} = 1 \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

To prioritize square detections near the center of the board, each detection mask $\mathbf{M}^{(k)}$ is weighted by a spatial confidence map $\mathbf{G}^{(k)} \in \mathbb{R}^{(n-k)\times(n-k)}$. The weights are defined by a normalized 2D Gaussian kernel centered at the midpoint of the matrix, and scaled by a factor 10^k to favor larger squares. Let $s = n - k$ be the kernel size and $(x_0, y_0) = \left(\frac{s+1}{2}, \frac{s+1}{2}\right)$ its center point, then the confidence map is given by:

$$G_{ij}^{(k)} = \frac{10^k}{N_g^{(k)}} \exp\left(-\frac{(i-x_0)^2 + (j-y_0)^2}{2\sigma^2}\right) \tag{17}$$

where $N_g^{(k)} = \sum_{i=1}^{n-k} \sum_{j=1}^{n-k} G^{(k)(i,j)}$ is the normalization factor and σ is set to unit. Hence, the reference square to be tracked can be defined as follows:

$$(k^\star, i^\star, j^\star) = \arg\max_{k,i,j} G^{(k)}_{(i,j)} M^{(k)}_{(i,j)} \tag{18}$$

As a result, this yields a square with the highest priority among all possible sizes and positions, where all four corner markers are detected.

2. Landing pad corner interpolation

After detecting the markers, a quadrilateral fitting step is applied to reduce tracking oscillations caused by the target's rolling motion. Unlike [3], the longest side of the quadrilateral is selected as the interpolation baseline, as this is the closest to the UAV along the vertical axis. A square is then fitted by rotating and adjusting it to minimize the absolute distances between its corners and the original points, as illustrated in Figure 3.

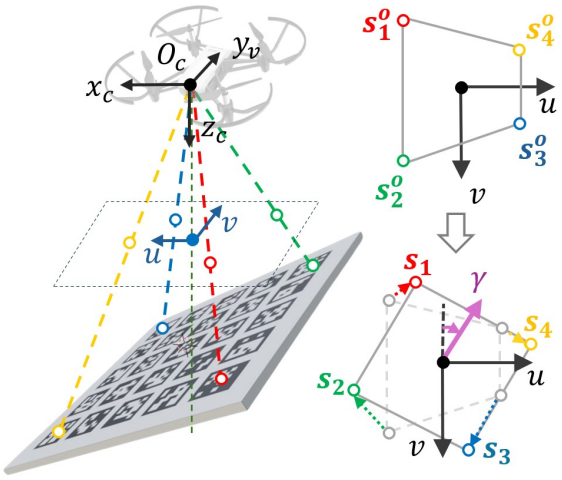

Fig. 3 Square interpolation procedure of detected image features.

B. Vessel state forecast

The relative target states are estimated using the standard Perspective-n-Point (PnP) method. The target attitudes relative to the world frame are computed since the UAV attitudes are available. The main idea of the prediction algorithm is to use an FFT approximation combined with a DKF similar to [9], hence it is referred to as FFT-DKF. The DKF can be introduced when the sampling rate of the vision pipeline is higher than that of the FFT window. Improvements in [12] were also included to decrease the number of high-value outliers. The target states within the previous time window ΔT_w can be modeled as shown in Eq. 19.

$$\eta(t) = \eta_0 + \sum_{i=1}^{n_f} \eta_i \cos\left(2\pi f_i t + \gamma_i\right), t \in [t_k, t_k + \Delta T_w] \tag{19}$$

where $\eta \in [X_t, Y_t, Z_t, \psi_t, \theta_t, \phi_t]$, and n_f is the total number of detected modes. A minimum threshold was applied,

defined as a proportion of the maximum amplitude within the current detection set, that is, $A_i \geq p \cdot \max_{k \in [1, n_f]} A_k$. This addresses measurement noise within the vision pipeline and the effects of UAV downwash on the landing pad. However, setting the threshold too high may exclude relevant modes that contribute to the motion dynamics. Based on experimental evaluation, a value of $p = 0.1$ was found to provide a good trade-off.

The parameters f_i, A_i, and ϕ_i will be updated when a time window is completed. Meanwhile, the vision pipeline still provides new pose, as the camera sampling rate is higher. Therefore, a Discrete Kalman Filter (DKF) estimates new amplitudes and phases, assuming that the modes remain the same. Once a new time window ΔT_w is complete, a couple of cases can occur: new frequencies may be detected, existing ones may disappear, or the same frequencies could reappear. Whenever the latter case occurs, the covariance matrices of the DKF are retained rather than reinitialized, with the initial state estimate based on the results from the previous window. The DKF model is represented in state space form in Eqs. 20 and 22.

$$\begin{aligned} \dot{\boldsymbol{\xi}}_\eta(t) &= \mathcal{A}\boldsymbol{\xi}_\eta(t), \\ \boldsymbol{\xi}_\eta(t) &= C\eta(t) \end{aligned} \tag{20}$$

where

$$\mathcal{A} = \begin{bmatrix} 0_{2,2} & 0_{2,2} & \dots & 0_{2,2} & 0_{2,2} \\ \boldsymbol{A}_1 & 0_{2,2} & \dots & 0_{2,2} & 0_{2,2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0_{2,2} & 0_{2,2} & \dots & \boldsymbol{A}_{n_f} & 0_{2,2} \end{bmatrix} \tag{21}$$

$$\boldsymbol{C} = \begin{bmatrix} \boldsymbol{C}_0 & \boldsymbol{C}_1 & \cdots & \boldsymbol{C}_{n_f} \end{bmatrix}$$

and

$$\boldsymbol{A}_i = \begin{bmatrix} 0 & 1 \\ -(2\pi f_i)^2 & 0 \end{bmatrix}, \quad \boldsymbol{C}_i = \begin{bmatrix} 1 & 0 \end{bmatrix}, \quad \boldsymbol{\xi}_\eta(t) = \begin{bmatrix} \boldsymbol{\nu}_0 \\ \boldsymbol{\nu}_1 \\ \vdots \\ \boldsymbol{\nu}_{n_f} \end{bmatrix}, \quad \boldsymbol{\nu}_i = \begin{bmatrix} \eta_i \cos(2\pi f_i t + \gamma_i) \\ -2\pi f_i \eta_i \sin(2\pi f_i t + \gamma_i) \end{bmatrix} \tag{22}$$

It should be noticed that $f_0 = 0$ and $\phi_0 = 0$, $\boldsymbol{\xi}_\eta \in \mathbb{R}^{(2n_f+1)\times 1}$, $A_i \in \mathbb{R}^{2\times 2}$, $\mathcal{A} \in \mathbb{R}^{2(n_f+1)\times 2(n_f+1)}$, $C_i \in \mathbb{R}^{2\times 1}$,

$C \in \mathbb{R}^{2(n_f+1)\times 1}$, and $\boldsymbol{\nu}_i \in \mathbb{R}^{2\times 1}$. Hence, the DKF will update the parameter whenever a sample is processed from the vision pipeline. The discrete model can be computed as follows considering $t = k\delta t + \Delta T_w$, where δt is the sampling time of the vision algorithm:

$$\begin{aligned} \dot{\boldsymbol{\xi}}_\eta^{(k+1)} &= e^{\mathcal{A}\delta t}\boldsymbol{\xi}_\eta^{(k)} = \mathcal{A}_d\boldsymbol{\xi}_\eta^{(k)} \\ \eta^{(k)} &= C\xi_\eta^{(k)} \end{aligned} \tag{23}$$

$$\begin{aligned} \hat{\boldsymbol{\xi}}^{(k)} &= \mathcal{A}_d\boldsymbol{\xi}^{(k-1)} & \boldsymbol{L}^{(k)} &= \hat{\boldsymbol{P}}^{(k)}C^T\left(C\hat{\boldsymbol{P}}^{(k)}C^T + \boldsymbol{R}\right)^{-1} \\ \boldsymbol{\xi}^{(k)} &= \hat{\boldsymbol{\xi}}^{(k)} + \boldsymbol{L}^{(k)}\left(\eta_m - \hat{\eta}^{(k)}\right) & \boldsymbol{P}^{(k)} &= \left(\boldsymbol{I} - \boldsymbol{L}^{(k)}C\right)\hat{\boldsymbol{P}}^{(k)} \\ \hat{\boldsymbol{P}}^{(k)} &= \mathcal{A}_d\boldsymbol{P}^{(k-1)}\mathcal{A}_d^T + \boldsymbol{Q} & \hat{\eta}^{(k)} &= C\hat{\boldsymbol{\xi}}^{(k)} \end{aligned} \tag{24}$$

where η_m is the measurement provided by the vision pipeline, $\boldsymbol{L} \in \mathbb{R}^{2(n_f+1)\times 1}$ is the Kalman gain, $\boldsymbol{P} \in \mathbb{R}^{2(n_f+1)\times 2(n_f+1)}$ is the state covariance matrix, $\boldsymbol{Q} \in \mathbb{R}^{2(n_f+1)\times 2(n_f+1)}$ is the covariance matrix of the process noise, and $\boldsymbol{R} \in \mathbb{R}$ is the measurement covariance. Following the procedure developed in [9], $\boldsymbol{R}$ is chosen as the mean amplitude of the measurement noise, and $\mathbf{Q}$ is initialized as $\mathbf{Q} = \frac{\lambda}{2}\left(\mathcal{A}\mathcal{A}^T + \mathbf{I}\right)\delta t$, where λ is a defined gain. The forecast step can be achieved by first retrieving the updated phase and amplitude for each mode as in Eqs. 25 and 26.

$$\Phi_i(t_m) = \gamma_i(t_m) + 2\pi f_i t_m = \arctan\left(\frac{2\pi f_i \boldsymbol{\nu}_{i,(1,1)}}{\boldsymbol{\nu}_{i,(1,1)}}\right) \tag{25}$$

$$\eta_i\left(t_m\right) = \frac{\mathbf{v}_{i,(1,1)}}{\sin\left(\Phi_i(t_m)\right)} \tag{26}$$

where t_m is the current time. Finally, the forecast model can be expressed as in Eq. 27.

$$\begin{aligned} \eta(t) &= \eta_0 + \textstyle\sum_{i=1}^{n}\eta_i(t_m)\cos\left(2\pi f_i(t-t_m) + \Phi_i(t_m)\right), t > t_m \\ \dot{\eta}(t) &= -\textstyle\sum_{i=1}^{n}2\pi f_i\eta_i(t_m)\sin\left(2\pi f_i(t-t_m) + \Phi_i(t_m)\right), t > t_m \end{aligned} \tag{27}$$

Overall, it is expected that the DKF will improve the quality of the forecast while new measurements are provided by the pose algorithm. This model also included the improvements introduced by [12]. The FFT was performed in overlapping time windows, using a discrete sliding function that prevents large discontinuities during the transitions. Hence, $\Delta t_{sliding}$ represents how many seconds from the previous time window will be included in the current FFT. Additionally, a sigmoid function is applied between consecutive FFT windows to ensure smooth mode transitions. This approach takes advantage of the DKF convergence from the previous step, while allowing enough time to integrate new measurements into the current window. The transition function used in this work is presented in Eq. 28.

$$\mathcal{K}(\delta t_e) = \sigma(c_1(\delta t_e - c_2)), \quad c_1 = \frac{10}{\Delta t_t}, \quad c_2 = \frac{\Delta t_t}{2} \tag{28}$$

where t_e is the relative time from the last timestamp when the FFT was triggered, c_1 regulates the speed of the transition, c_2 is an offset term, and Δt_t is the sigmoid transition time window.

In the experimental setup, a data quality check was introduced to address conditions where the vision system was unable to detect all feature points. A quality index was defined as the ratio of valid to total samples. This index was computed for both the entire FFT window and the final 25% of the window to take into account more recent measurements. The thresholds were set at 90% for the entire window and 50% for the final quarter. If either of these thresholds was not met, the DKF was re-initialized, as the data were deemed unreliable and potentially outdated.

The FFT-DKF approach relies on experimental tuning as presented in the accuracy study in [12]. However, a large-scale vessel will predominantly act as a low-pass filter of the random-wave input, so high-frequency terms are less important within the investigated scenarios. Although this forecast method does not require prior knowledge of the vessel's dynamics, it relies on a linear approximation. As a result, its accuracy is restricted to short-term forecasts [9], as will be demonstrated.

C. VS-NMPC formulation

The proposed VS-NMPC is based on the visual servoing dynamics introduced in Eq. 14. The inner UAV controller is assumed to track the input speed $\boldsymbol{V}_r^v = [v_x^v, v_y^v, v_z^v, \omega_z^v]^T$. To simplify the VS-NMPC notation, $\boldsymbol{\mu} = \boldsymbol{V}_r^v$. The discrete time state space of Eq. 14 is then given by

$$\boldsymbol{q}_{n+1} = \mathbf{f}_{d,\Delta t}\left(\boldsymbol{q}_n, z^\star, \boldsymbol{\mu}_n\right) \tag{29}$$

where n represents the time step, and $f_{d,\Delta t}$ is the discrete servoing dynamics at a sampling rate Δt. The image moment errors are defined as $\tilde{\boldsymbol{q}}_n = \boldsymbol{q}_n - \boldsymbol{q}_{n,ref}$. For the nonlinear optimization framework, the objective function will include the standard quadratic terms for both image moment error and speed inputs. Inspired by [9], a barrier function is proposed to include the states of the vessel. Therefore, the complete cost term is outlined in Eq. 30.

$$\min_{\tilde{\mu}_1,\ldots,\tilde{\mu}_{N-1}} V(\tilde{\boldsymbol{q}}, \tilde{\boldsymbol{\mu}}) = \underbrace{\sum_{n=1}^{N} \tilde{q}_n^T \mathbf{Q}_q \tilde{q}_n + \sum_{n=1}^{N-1} \tilde{\mu}_n^T \mathbf{Q}_\mu \tilde{\mu}_n}_{J_1} \quad + \underbrace{\sum_{n=1}^{N} \boldsymbol{\alpha}_g \cdot \boldsymbol{g}_b\left(q_{h_n}, z^\star, \boldsymbol{\eta_n}\right)}_{J_2}, \tag{30}$$

subjected to:

$$\tilde{\boldsymbol{q}}_n = \boldsymbol{\omega_q} \odot (\boldsymbol{q}_n - \boldsymbol{q}_{n,\text{ref}}), \tag{31}$$

$$\boldsymbol{q}_{n+1} = \mathbf{f}_{d,\Delta t}\left(\boldsymbol{q}_n, z^\star, \boldsymbol{\mu}_n\right), \tag{32}$$

$$\tilde{\boldsymbol{\mu}}_n = \boldsymbol{\omega_\mu} \odot \boldsymbol{\mu}_n, \tag{33}$$

$$\boldsymbol{q}_n \in \mathcal{S} = \left\{\boldsymbol{q} \in \mathbb{R}^{4\text{x}1} : \mathbf{q}_{\min} \leq \boldsymbol{q} \leq \mathbf{q}_{\max}\right\}, \tag{34}$$

$$\boldsymbol{\mu}_n \in \mathcal{U} = \left\{\boldsymbol{\mu} \in \mathbb{R}^{4\text{x}1} : \boldsymbol{\mu}_{\min} \leq \boldsymbol{\mu} \leq \boldsymbol{\mu}_{\max}\right\}. \tag{35}$$

where N is the prediction horizon, $\mathcal{U}$ and $\mathcal{S}$ represent the constraint sets for the image moments and speed inputs, $\boldsymbol{\omega_q}$ and $\boldsymbol{\omega_\mu}$ are optimization preconditioning scaling for states and input vectors, $\mathbf{Q}_q \in \mathbb{R}^{4\text{x}4}$ and $\mathbf{Q}_\mu \in \mathbb{R}^{4\text{x}4}$ are diagonal weighting matrices, $\boldsymbol{g_b}$ is the barrier function, and $\boldsymbol{\alpha}_g$ its weight factor. Hence, J_1 corresponds to the traditional quadratic term, and J_2 is the additional cost term that takes into account the target states.

For this paper, only the target roll angle is considered; therefore, $\boldsymbol{\alpha}_g$ is actually a scalar weight α_g, $\boldsymbol{\eta}_n$ corresponds to the roll angle ($\boldsymbol{\eta}_n = \phi_{t,n}$), and $\boldsymbol{g}_b$ is a scalar function $g_b\left(q_{h_n}, z^\star, \phi_{t,n}\right)$. Moreover, the prediction and control horizons are the same given the short-term forecast nature of the problem. The main goal of the barrier function is to prevent touchdown in conditions that the forecasted roll angle of the landing pad are excessively high. However, for a given forecast set, the associated cost should be weighted by the relative height, as the UAV is still expected to initiate a descent if it is significantly above the landing pad. Assuming that the target does not have heave motion, the relative height can be directly computed as $z = z^\star q_h$. Finally, the proposed barrier function g_b is defined in Eqs. 36 and 37.

$$\boldsymbol{g}_b(z, \phi_{t,n}) = f(z) \cdot (\phi_{t,n})^2 \tag{36}$$

$$f(z) = \underbrace{\sigma\left(\left(z_{d,1} - z\right) a_1^{-1}\right)}_{\sigma_1} + \underbrace{\sigma\left(\left(z_{d,2} - z\right) a_2^{-1}\right) - 1}_{\sigma_2} + f_0 \tag{37}$$

Fundamentally, this function represents two sigmoid components with transition heights $z_{d,1}$ and $z_{d,2}$, while α_1 and α_2 control their steepness. Note that $z_{d,1} > z_{d,2}$, $a_1 > 0$ and $a_2 < 0$. The parameter $f_0 > 0$ is introduced to ensure that f remains positive due to the properties of σ_2. However, J_0 should be kept sufficiently small to avoid significantly affecting the behavior of the overall cost function. As a result, Eq. 37 does not introduce discontinuities like the function proposed in [9]. Furthermore, as presented in Eq. 30, the combined sigmoid functions in Eq. 37 are weighted by both the forecast roll angle and the scalar weight α_g. This approach ensures that the cost function escalates when high roll angles are detected, effectively preventing the UAV from descending below the threshold height z_{ref}. Therefore, the steepness controlled by α_2 will actually define the extension of the waiting region, while α_1 the aggressiveness of the

landing. By properly tuning the relationship between α_g and the other weight matrices, $\mathbf{Q}_q$ and $\mathbf{Q}_\mu$, these parameters will define the output landing envelope when it comes to the maximum allowed roll angle of the platform during touchdown. An example of $f(z)$ function is given in Fig. 4.

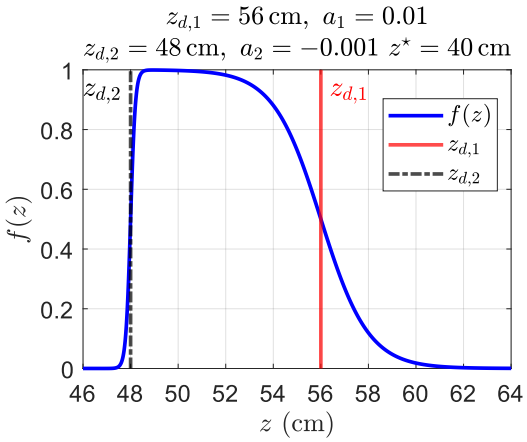

Fig. 4 Example of VS-NMPC cost barrier function.

A feedforward term based on the estimated horizontal speeds of the landing pad is added to the optimization result $\boldsymbol{\mu}_1$ in a similar approach as [3]. In a minimal cooperative scenario, it is assumed that the UAV has access to the current speed of the target. To simulate this setup, the target position is acquired using an Optical Motion System (OMS), as illustrated in Fig. 2. Given the high sampling rate, a Kalman Filter (KF) with a constant velocity model is applied to estimate the target horizontal speed. These estimated velocities are then transformed into the virtual camera frame, resulting in the velocity vector $\boldsymbol{\mu}_{FF} = [v^v_{x,ff}, v^v_{y,ff}, 0, 0]^T$. Note that Euler angles of both the target and the UAV with respect to the world frame are also available from the OMS. The input speed for the inner UAV controller is given by $\boldsymbol{\mu}_1 + \boldsymbol{\mu}_{FF}$.

IV. Test setup

The hardware setup for the UAV and landing target are described in the Subsection IV.A. In Subsection IV.B, the parameters of the landing pad model are presented. The RAOs used to simulate the vessel's oscillations were those of the destroyer-class USS DLG-26, as reported in [17]. The sea state conditions and wave spectra parameters were the values from the North Atlantic Sea database, as provided by [1]. In Subsection IV.C, the scale-down landing profile is discussed along with the criteria for a successful landing. The roll constraints for touchdown used in this paper follow the daytime operational limitations shown in [18]. For flight test, touchdown precision was assessed by scaling down the tolerances for the MQ-8B rotocraft [19]. In [19], the maximum touchdown error is ± 3.5 ft, which is equivalent to ±12.5% of its rotor diameter. Finally, in Subsection IV.D, both simulation and flight scenarios are described given the conditions and parameters for each test.

A. Hardware Setup

The experimental setup consisted of a Ryze Tello UAV as detailed in Fig.5. The speed calibration curves were based on the previous work of the authors [12]. For flight tests, the speed commands were processed externally. The landing pad motion was simulated using a robotic arm installed on top of a UGV, as presented in Fig.5. On-board processing using a simple inverse kinematics provided inputs to the 4 DoF robotic arm to track the SS 6 roll motion. The VICON OMS used is described in [20].

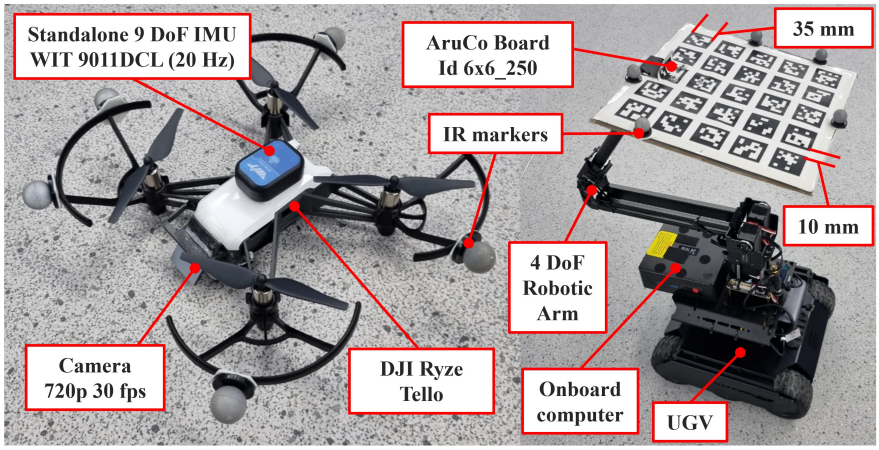

Fig. 5 Description of experimental hardware (UAV and landing pad).

B. Target motion parameters

The parameters of the Bretschneider spectra for $SS = 6$ are $H_{1/3} = 5.0\,m$ and $\omega_m = 0.51\,rad/s$. The simulated scenario included $U = 10\,kt$, $\chi = 150°$, considering the motion of the center of the deck. Simulation results comparing mild (SS 3) and severe conditions (SS 6) are given in Fig. 6. It can be seen that heave and roll oscillations are the most relevant states in harsh sea scenarios. In this paper, only roll motion of the deck was investigated as an initial proof of concept for the proposed controller.

C. Scaled-down Landing profile

Three approaches were tested: (i) baseline IBVS from literature that includes a landing criteria based on current target states, (ii) IBVS with landing criteria based on forecasted target using FFT-DKF, in short IBVS + FFT-DKF, and (iii) VS-NMPC. The landing procedure is described in Fig. 7 for each one. A simulated landing height of 46 cm above the board was selected due to the limitations of the UAV's LiDAR and optical flow systems. At this height, the UAV downwash has minimal impact on the motion of the robotic arm. On the other hand, both ground effect and camera FoV constraints at low height are not considered. The decision height means that the UAV waits until the conditions are met to attempt the landing. The VS-NMPC has a waiting region due to the nature of the custom barrier function.

The landing profile was classified into three stages as shown in Fig. 7: (I) high hover tracking, (II) low hover

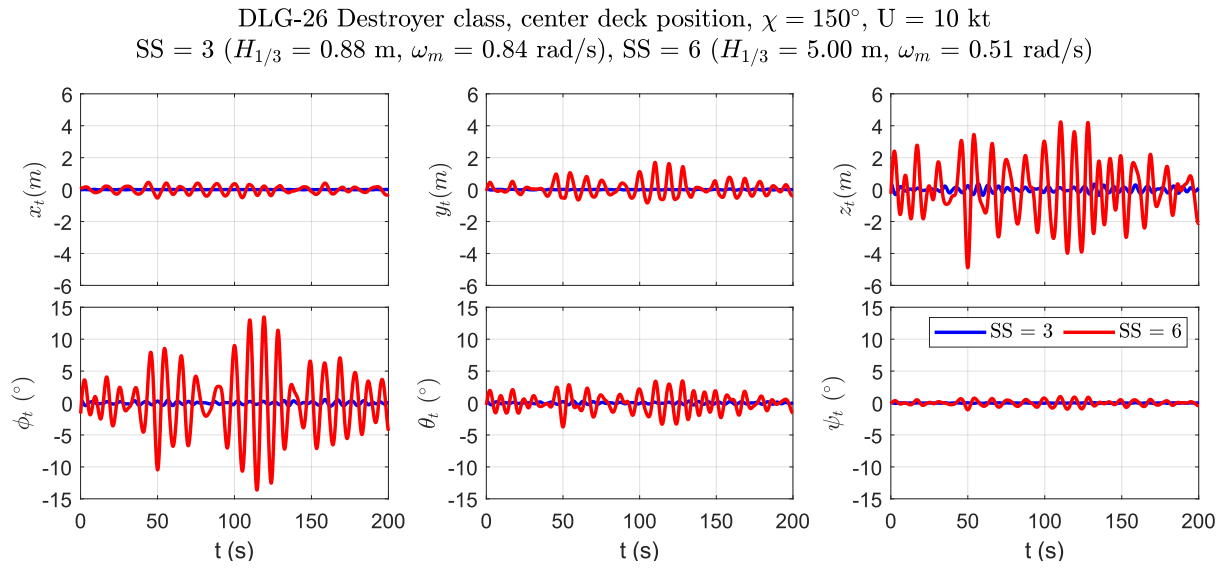

Fig. 6 Comparison of landing deck oscillation under mild (SS 3) and harsh (SS 6) sea states.

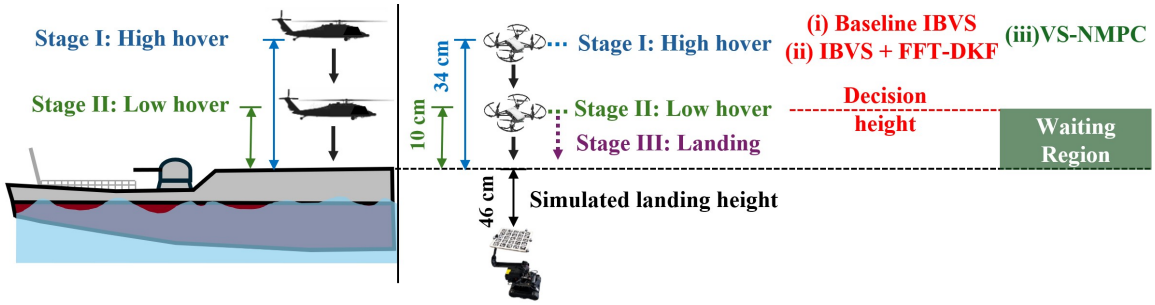

Fig. 7 Scaled-down landing profile for (i) baseline IBVS, (ii) IBVS + FFT, and (iii) VS-NMPC.

tracking, and (III) landing attempt. For both baseline IBVS and IBVS + FFT-DKF, the landing decision was triggered while tracking the low hover position. The VS-NMPC has a waiting region that is defined based on the barrier function presented in Fig. 4 hence stages (II) and (III) are combined.

Table 1 summarizes the conditions to trigger the transition between stages, and the tracking references. For the baseline IBVS and IBVS + FFT+DKF, reference heights for stages I and II, $z_I^\star$ and $z_{II}^\star$) respectively, are detailed in Table 1. During stage III these controllers track a given rate of descent (V_z^v). Therefore, $z_{II}^\star$ was defined to prompt touchdown. An additional criteria based on the tracking error of the features ($\delta\epsilon_i$) was introduced to manage the transitions between stages. Therefore, $\delta\epsilon_i = ||\boldsymbol{s}_i^v - \boldsymbol{s}_{i,ref}^v||$ is the i-th feature tracking error, while $\boldsymbol{s}_{i,ref}^v$ is the i-th tracking references on the virtual image plane.

Both touchdown precision and platform roll angle were used to assess landing performance. A landing attempt was considered successful when the UAV achieved the simulated landing height when $|\phi_t| \leq 6°$. Regarding precision, the results were compared with touchdown tolerances of ±12.5% and ±25.0% of the equivalent rotor diameter of the quadcopter. The equivalent rotor diameter was defined as $D_{eq} = \sqrt{N_{rotors}} D_{rotor}$, where N_{rotors} is the number of

Table 1 Transition conditions between landing profile stages for each controller

Stage	Baseline IBVS	IBVS + FFT-DKF	VS-NMPC
(I) to (II)	$z_I^\star = 80\,\text{cm}$, $\delta\epsilon_i < 40$ for 3 s		
(II) to (III)	$z_{II}^\star = 56\,\text{cm}$, $\delta\epsilon_i < 40$ for 3 s, $\lvert\phi_{t,n}\rvert < 6^\circ$ for $n = 0$	$z_{II}^\star = 56\,\text{cm}$, $\delta\epsilon_i < 40$ for 3 s, $\lvert\phi_{t,n}\rvert < 6^\circ$ for $n \in [0, N]$	Cost definition and $z_{II}^\star = 40\,\text{cm}$
(III)	$V_z^v = 14\,\text{cm/s}$	$V_z^v = 14\,\text{cm/s}$	

rotors and D_{rotor} the diameter of the individual rotors.

D. Simulation and flight test scenarios

Throughout all simulations and flight tests, the FFT-DKF was set with the following parameters: $\Delta T_w = 20\,s$, $\lambda = 5$, $R = 1^\circ$, $\Delta t_t = 2\,s$, $\delta t_{sliding} = 10\,s$, $p = 0.1$, and $\delta t = 0.05\,s$. These parameters follows the findings in [12], where predictions with $\Delta T_w = 20\,s$ were more accurate than $\Delta T_w = 10\,s$. Moreover, the vision pipeline runs on average at 20 Hz, while the FFT updates at 0.1 Hz. This sampling rate difference supports the introduction of the DKF. For the VS-NMPC, the IPOPT NLP solver [21] is used with direct multiple shooting and Euler integration. The barrier function is the same as described in Fig. 4, with $f_0 = 2.0 \times 10^{-4}$. The sampling rate was $\Delta t = 0.1\,s$, and the cost parameters were $\alpha_g = 15/N$, $\boldsymbol{Q}_q = 1.0 \times 10^3\,\mathbb{I}_4/N$, and $Q_{\mu,v_x^v} = Q_{\mu,v_y^v} =, Q_{\mu,\omega_z^v} = 10$ where $\mathbf{Q}_\mu = diag(Q_{\mu,v_x^v}, Q_{\mu,v_y^v}, Q_{\mu,v_z^v}, Q_{\mu,\omega_z^v})/N$. The scaling vectors were given by $(\boldsymbol{\omega}_\mu)_j = 1/(\boldsymbol{\mu}_{max})_j$, and $\boldsymbol{\omega}_q = [1, 1, 1, 1]^T$. The state constraints were set based on the intrinsic camera parameters that resulted in $\mathbf{q}_{\min} = [-2.57, -1.98, 0, -3.14]^T$, and $\mathbf{q}_{\max} = [2.57, 1.98, 3, 3.14]^T$. The input constraints that were common across simulations and flight tests were $\mu_{V_x^v}, \mu_{V_y^v} \in [-5, 5]\,cm/s$, and $\mu_{\omega_z^v} \in [-25, 25]^\circ/s$.

The following subsections describe the specific conditions and parameters for simulations and flight tests.

1. Simulation conditions

For simulations, a simplified UAV speed model from [22] is used. Each simulation run had a flight time of 40 min, and the UAV was repositioned at $z_I^\star$ when it crossed the simulated landing height. First, a tuning investigation provided landing results considering PH values of $1.0\,s$, $1.5\,s$ and $2.0\,s$ combined with Q_{μ,v_z^v} of 50 and 200. This investigation also covered the vertical speed constraints $\mu_{V_z^v} \in [-4, 14]\,cm/s$ and $\mu_{V_z^v} \in [-4, 8]\,cm/s$. Next, the scenarios in Table 2 were used to compare the controllers.

2. Flight test conditions

During the flight tests, after each simulated landing, the UAV returned to stage I (high hover tracking) to attempt another landing. Four different scenarios were tested, as shown in Table 3. Gain values (λ) for the baseline IBVS and the IBVS + FFT-DKF controllers were tuned through experiments. All other parameters matched those used in the simulations.

Table 2 Description of simulation scenarios.

Scenario	Controller	Target motion	PH (s)	Main parameters
SIM-a	IBVS	Roll (SS 6)	-	$\lambda = diag(0.35, 0.35, 0.35, 1)$
SIM-b	IBVS + FFT-DKF	Roll (SS 6)	2.0	$\lambda = diag(0.35, 0.35, 0.35, 1)$
SIM-c	VS-NMPC	Roll (SS 6)	2.0	$Q_{\mu,v_z^v} = 200$, $\mu_{min,V_z^v} = -4\,\text{cm/s}$, and $\mu_{max,V_z^v} = 14\,\text{cm/s}$

Table 3 Description of flight test scenarios.

Scenario	Controller	Target motion	PH (s)	Main parameters
FT-a	IBVS + FFT-DKF	Roll (SS 6)	2.0	$\lambda = diag(1.5, 1.5, 0.35, 1)$
FT-b	VS-NMPC	Roll (SS 6)	2.0	$Q_{\mu,v_z^v} = 50$ $\mu_{max,V_z^v} = 14\,cm/s$
FT-c	VS-NMPC	Roll (SS 6)	1.0	
FT-d	VS-NMPC	Roll (SS 6), and $U = 6\,cm/s$	2.0	

V. Results

This section is divided into two parts: the first presents the outcomes of the simulations, while the second focuses on the flight test results. The simulation study evaluates the performance of the VS-NMPC under varying prediction horizons, optimization constraints, and weight costs. This initial study provides the necessary settings to compare the VS-NMPC with the IBVS controllers. Although simulations only consider landing performance based on the target roll constraint, indoor flight tests also introduce the touchdown accuracy criteria. Flight tests also covered the results of on-line processing under a wide range of test conditions that include roll and forward motion of the target.

A. Simulation Outcomes

First the VS-NMPC is tested with different values of PH, μ_{max,V_z^v}, and Q_{μ,v_z^v} to tune the controller. The results are detailed in Table 4 where η_{L_ϕ} is the landing success rate based on the target roll envelope ($\pm 6°$) , N_L is the total number of landing attempts, and APM is the number of attempts per minute.

In Table 4, η_{L_ϕ} decreases as PH increases for lower descent speed ($\mu_{max,V_z^v} = 8\,cm/s$) and $Q_{\mu,v_z^v} = 50$. In general, this is related to poorer forecast quality for higher PH. This η_{L_ϕ} degradation can be mitigated by increasing Q_{μ,v_z^v} to 200 that lowers the control bandwidth; however, this also decreases the APM. However, if the descent rate constraint is increased to $\mu_{max,V_z^v} = 14\,cm/s$, a trade-off can be achieved in terms of η_{L_ϕ} performance, but with higher landing attempts than $Q_{\mu,v_z^v} = 50$.

As a result, the VS-NMPC configuration with $PH = 2.0\,\text{s}$, $Q_{\mu,v_z^v} = 200$, $\mu_{min,V_z^v} = -4\,\text{cm/s}$, and $\mu_{max,V_z^v} = 14\,\text{cm/s}$ was selected for comparison with the baseline IBVS and IBVS + FFT-DKF approaches. This choice ensures that all methods operate under the same maximum descent speed, and that both VS-NMPC and IBVS + FFT-DKF share

Table 4 Tuning results for VS-NMPC simulations.

PH (s)	Q_{μ,v_z^v}	$\mu_{min/max}$ V_z^v (cm/s)	N_L	η_{L_ϕ} (%)	APM
1.0	50	-4 / 8	157	93.0	3.9
1.5	50	-4 / 8	192	87.5	4.8
2.0	50	-4 / 8	231	77.5	5.8
1.0	200	-4 / 8	77	98.7	1.9
1.5	200	-4 / 8	78	96.2	1.9
2.0	200	-4 / 8	95	93.7	2.4
1.0	50	-4 / 14	313	83.4	7.8
1.5	50	-4 / 14	307	88.3	7.7
2.0	50	-4 / 14	314	88.5	7.8
1.0	200	-4 / 14	201	97.5	5.0
1.5	200	-4 / 14	201	91.0	5.0
2.0	200	-4 / 14	251	94.8	6.3

the same prediction horizon. Given the definitions of $z_{II}^{\star}$ and the waiting region, the starting position to attempt the landing is approximately the same; thus, a similar time to reach the simulated landing height.

Figure 8 presents both UAV height and target roll for each controller approach under the same sea state database. The plot shows a sample of the first 550 seconds. For instance, the VS-NMPC approach successfully avoids landing in high sea state between 200 s and 270 s.

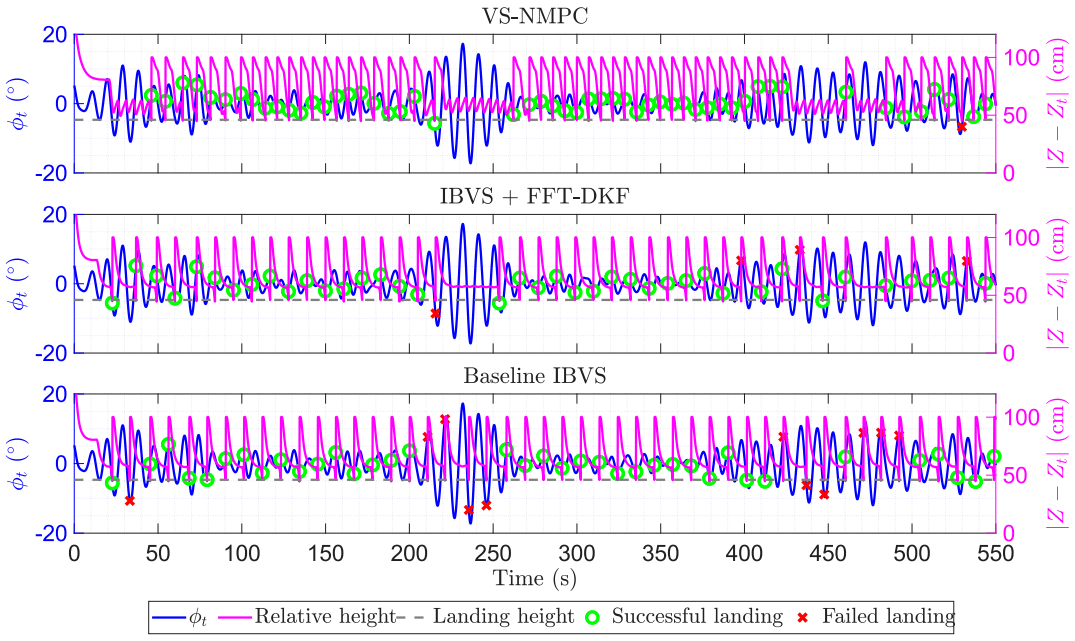

Fig. 8 Vertical profile sample comparing VS-NMPC, IBVS + FFT-DKF, and baseline IBVS.

The VS-NMPC cost values are detailed in Fig. 9 for the first 350 s. This figure shows that the UAV moves towards

landing height only when J_1 and J_2 decrease simultaneously.

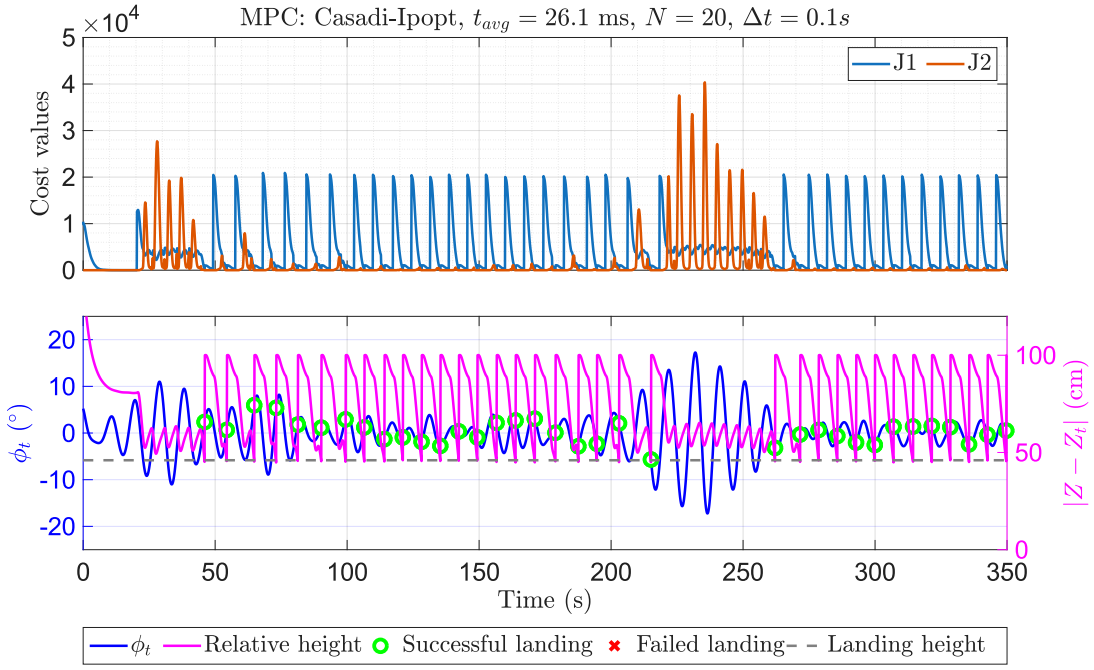

Fig. 9 Vertical profile results for tuning condition $PH = 2.0\,s$, $Q_{\mu,v_z^v} = 200$ and $\mu_{max,V_z^v} = 14\,cm/s$.

The results comparing the approaches for the entire sea state database during the 40 min flight time are summarized in Table 5. The baseline IBVS controller has the lowest landing success rate ($\eta_{L,\phi}$). This highlights that the landing criteria based on the current target states is unsafe under harsh sea state conditions. By introducing the forecast states, the IBVS + FFT-DKF improves $\eta_{L,\phi}$, although the number of landing attempts decreases. Finally, the VS-NMPC yields the highest rate of successful touchdowns, while also increasing the number of attempts per minute.

Table 5 Overall landing performance for simulation scenarios.

Parameter	Baseline IBVS (SIM-a)	IBVS + FFT-DKF (SIM-b)	VS-NMPC (SIM-c)
N_L	213	197	251
η_{L_ϕ} (%)	67.9	82.7	94.8
APM	5.3	4.9	6.3

B. Flight Test Results

Given the poor simulation results of the baseline IBVS, the flight tests included only the IBVS + FFT-DKF and VS-NMPC. The conditions for each flight test scenario are detail in Table 3. A sample of the vertical profile and cost values for scenario FT-b is shown in Fig. 10. First, ϕ_t obtained from the vision pipeline show overall good agreement with those from the OMS. Moreover, Fig. 10 demonstrates that the VS-NMPC triggers the landing only when both J1

and J2 terms decrease.

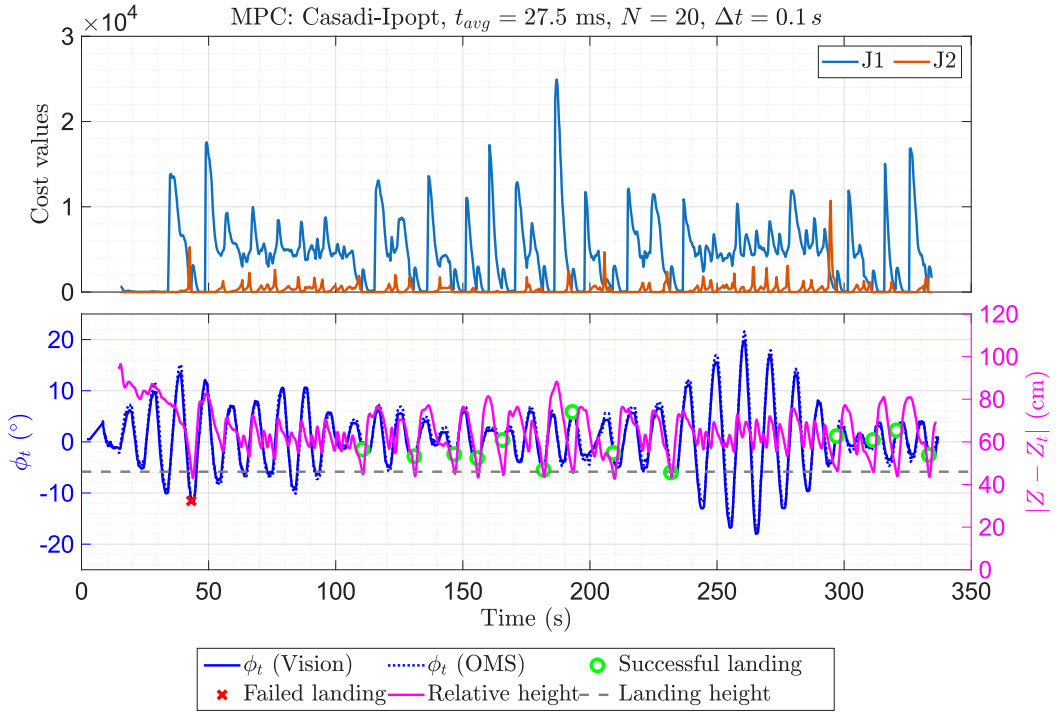

Fig. 10 Sample of vertical profile results for scenario FT-b.

Detailed results of the ϕ_t distribution during touchdown are presented in Fig. 11, while precision is shown in Fig. 12 where both precision tolerances are highlighted (12.5%D_{eq} and 25.0%D_{eq}). In Fig. 12, δx_t and δy_t are expressed as a percentage of D_{eq}, they are the longitudinal and lateral deviations from the center of the landing pad on the target body frame. In addition, the 90% Circular Error Probable (CEP) radius is shown for comparison.

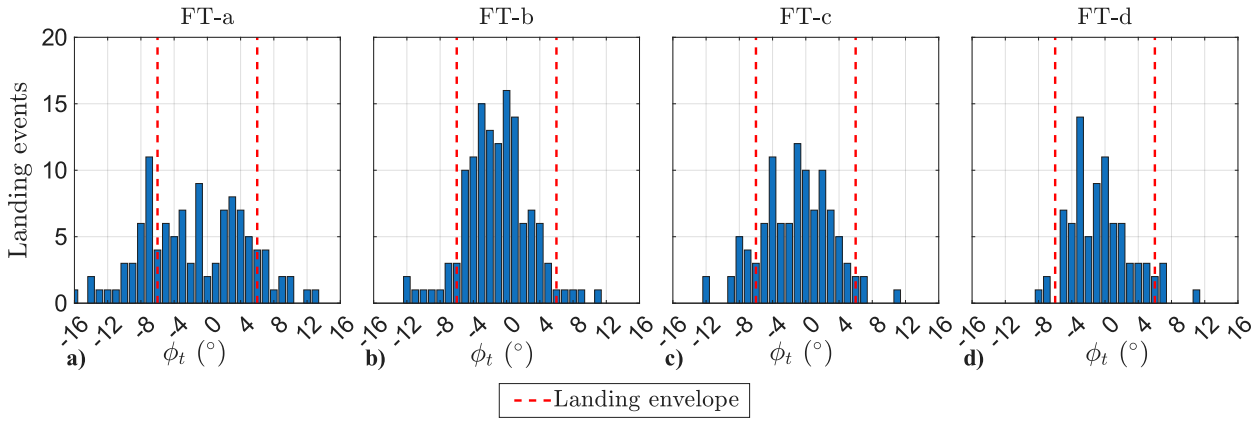

Fig. 11 ϕ_t during landing for flight test scenarios: a) FT-a, b) FT-b, c) FT-c, and d) FT-d.

The final results are summarized in Table 6, where Δt_{FT} is the total flight time, $\eta_{\mathbf{L}_\phi}$ is the landing success rate based only on the roll envelope, and $\eta_{\mathbf{L}_{\phi,12.5\%}}$ and $\eta_{\mathbf{L}_{\phi,25.0\%}}$ include both roll and touchdown tolerances of ±12.5% D_{eq} and

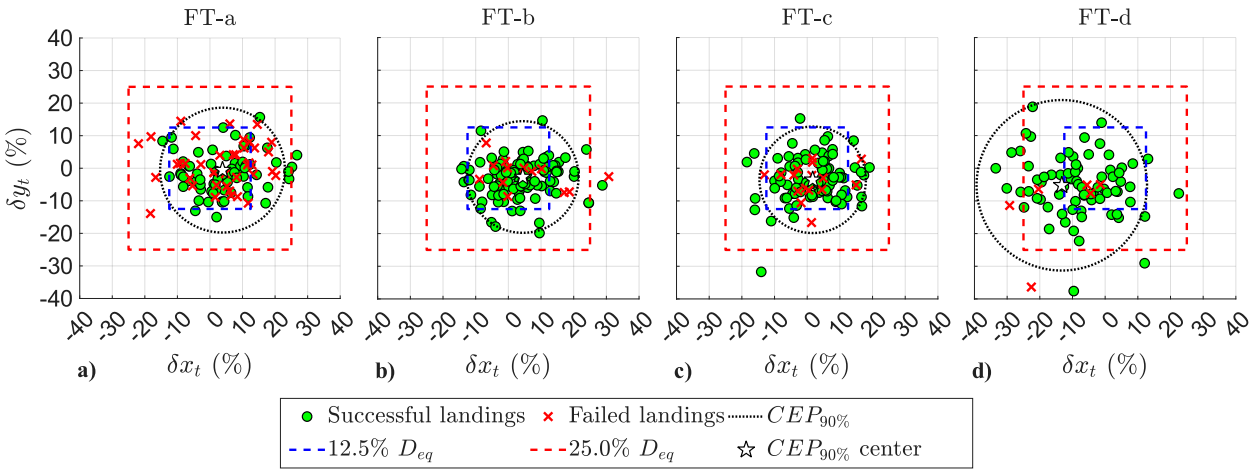

Fig. 12 Landing precision for flight test scenarios: a) FT-a, b) FT-b, c) FT-c, and d) FT-d

±25.0% D_{eq}, respectively. A two-sample Kolmogorov–Smirnov (KS) test at a 5% significance level, comparing the full roll dataset and the roll values at landing, yields p-values that lead to rejection of the null hypothesis. This indicates that the two distributions differ, confirming that the landing outcomes were not biased by the target roll distribution.

Table 6 Overall landing performance for flight test scenarios.

Scenario	PH (s)	N_L	Δt_{FT} (min)	APM	η_{L_ϕ} (%)	$\eta_{L_{\phi,25.0\%}}$ (%)	$\eta_{L_{\phi,12.5\%}}$ (%)	Two-sample KS test Result	Two-sample KS test p-value
FT-a	2.0	110	44	2.5	63.6	61.8	51.8	Rejected	3.55×10^{-3}
FT-b	2.0	130	54	2.4	90.0	88.5	66.2	Rejected	1.28×10^{-5}
FT-c	1.0	104	44	2.4	84.6	83.7	69.2	Rejected	3.56×10^{-5}
FT-d	2.0	82	44	1.9	91.5	81.7	42.7	Rejected	1.02×10^{-3}

As in the simulations, increasing PH to 2.0 s improved the landing success η_ϕ without affecting the number of landing attempts per minute (APM). This improvement is evident when comparing scenarios FT-b and FT-c in Table 6. Furthermore, $\eta_{\phi,25.0\%}$ still shows a slight increase with longer PH. On the other hand, $\eta_{\phi,12.5\%}$ decreases, indicating that the controller struggles to consistently meet the most stringent landing precision requirements. When comparing the VS-NMPC (FT-b) with the IBVS + FFT-DKF (FT-a) under the same PH settings, there is a significant improvement across landing success rates, while APM remains largely unaffected. The differences on the ϕ_t distributions in Fig. 11(a) and Fig. 11(b) show that IBVS + FFT-DKF fails to consistently satisfy the roll constraints during touchdown. Introducing a feedforward term (comparing FT-a and FT-d) maintains a stable value of η_{L_ϕ}; however, it leads to a decrease in landing precision, particularly for $\eta_{\phi,12.5\%}$. This degradation is also reflected in the $CEP_{90\%}$ metrics by comparing Fig. 12(b) with Fig. 12(d). Furthermore, the APM slightly deteriorates as the forward speed of the target increases. Furthermore, Fig. 13 demonstrates how inaccurate forecasting results in landing failures. This plot shows forecast quality before all touchdown points in FT-d. Two failed landing attempts labeled 1 and 2 are used as an example.

It is shown that 2.0 s prior to landing, the forecast predicted roll angles near zero despite actual values exceeding constraints. Although forecast accuracy improved as the UAV approached touchdown, the barrier function had already been breached.

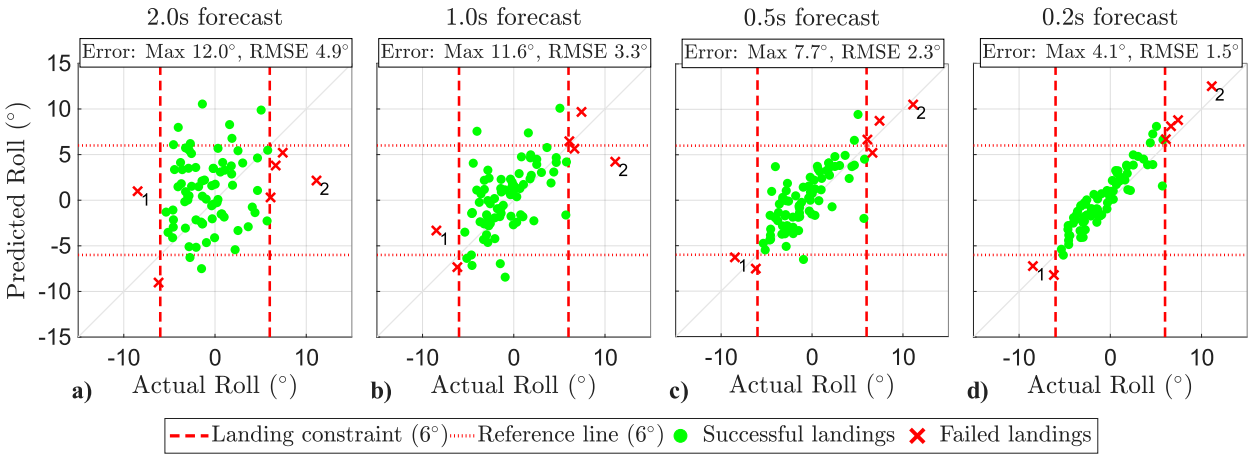

Fig. 13 Forecast quality before touchdown for FT-d with forecasts: a) 2.0 s, b) 1.0 s, c) 0.5 s, and d) 0.1 s

The forecast performance under partial occlusions is shown in Fig. 14. There is no major degradation when subjected to minor occlusion. Even tought uncertainty increases following the DKF-FFT windown transitions, the DKF performs the corrections under minor occlusions as seen between 65 and 85 s. On the other hand, minor and followed by major occlusions after the window transition introduced error after 105 s. It should be noticed that during the flight tests database no major occlusion was experienced.

The forecast performance under partial occlusions is shown in Fig. 14. There is no major degradation under minor occlusions. Although uncertainty increases following the DKF-FFT window transitions, the DKF successfully performs corrections under minor occlusions, as observed between 65 and 85 s. However, minor occlusions followed by major occlusions introduce errors as seen after 105 s. It should be noted that this test included manual occlusions of multiple markers.

Overall, the results demonstrate the feasibility of the VS-NMPC approach under varying prediction horizons and online conditions. Moreover, it shows the improvement in safe landing rates of the VS-NMPC controller compared to an enhanced version of the IBVS used in literature. However, heave oscillations are not covered within this test scenarios. Since the UAV state estimator is not customizable, the optical flow sensor would instead provide relative vertical speed. Scaling-down both vessel motion and UAV bandwidth is required to achieve generalization to operational decks [10]. This would allow the simulated landing height to be decreased, which will also require addressing camera FoV constraints during low hover. Furthermore, wind turbulence is critical in high sea states, yet scaling down these disturbances remains challenging given the interaction with the vessel structures. One potential solution is to include a scaled-down approach of Control Equivalent Turbulence Inputs (CETI) [23].

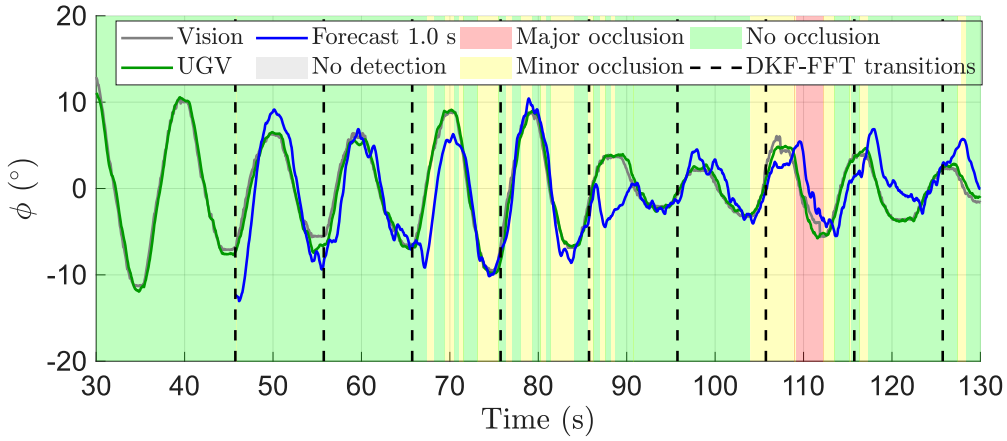

Fig. 14 DKF-FFT performance under occlusions.

VI. Conclusion and future works

This paper presents a visual servoing nonlinear model predictive control (VS-NMPC) for autonomous UAV landing on vessels under harsh sea oscillations. The vision pipeline performs feature detection and target pose estimation. Future target states are forecasted using an FFT-DKF algorithm. These states are introduced into cost barrier functions to ensure that touchdowns occur when the target's roll angle is within 6°. Simulations show that the VS-NMPC achieves 94.8% landing success rate under SS 6, while baseline IBVS and an enhanced IBVS with the forecasted states (FFT-DKF IBVS) reach 67.9% and 82.7%. Indoor flight tests using a small-scale UAV and a landing pad mounted on a robotic arm provide results under online processing conditions. An additional landing constraint was introduced for flight tests considering a touchdown tolerance of ±25.0% of the equivalent rotor diameter. In the scenario with landing pad roll oscillations, the VS-NMPC achieves 88.5% success against 61.8 % of the FFT-DKF IBVS. If target forward speed is included, the VS-NMPC success rate remained at 81.5%. These results highlight not only the advantages of the VS-NMPC in terms of safe landings under challenging sea environments, but also the shortcomings of conventional IBVS in the same scenario.

Flight tests also revealed limitations of the VS-NMPC with the feedforward term, as the touchdown error increased even at low target forward speeds. One potential solution is to integrate the vessel state feedback directly into the VS-NMPC to better anticipate target motion. In addition, heave motion plays a critical role in rough sea conditions. When target heave oscillations are included, the current barrier function may trigger landing prematurely when large height-tracking errors occur. Furthermore, extending the results to full-scale systems requires proper scaling of control bandwidth, vessel motion, and UAV inertia to accurately evaluate rollover mitigation. Although this would require changes to the UAV inner attitude controller, it also opens opportunities to properly address camera field-of-view constraints.

Acknowledgments

This work was supported and sponsored by MBDA UK and the Brazilian Air Force through sponsorship P20227.

References

[1] BALES, S. L., "Designing Ships to the Natural Environment," Naval Engineers Journal, Vol. 95, No. 2, 1983, pp. 31–40. https://doi.org/10.1111/j.1559-3584.1983.tb00574.x.

[2] (WMO), W. M. O., "Guide to Instruments and Methods of Observation, Volume I – Measurement of Meteorological Variables (WMO-No. 8)," Technical report, WMO - World Meteorological Organization, 2025. https://doi.org/10.59327/WMO/CIMO/1.

[3] Cho, G., Choi, J., Bae, G., and Oh, H., "Autonomous Ship Deck Landing of a Quadrotor UAV using Feed-forward Image-based Visual Servoing," Aerospace Science and Technology, Vol. 130, November, 2022, p. 107869. https://doi.org/10.1016/j.ast.2022.107869.

[4] Wynn, J. S., and McLain, T. W., "Visual Servoing with Feed-Forward for Precision Shipboard Landing of an Autonomous Multirotor," 2019 American Control Conference (ACC), Philadelphia, PA, USA, 2019, pp. 3928–3935. https://doi.org/10.23919/ACC.2019.8814694.

[5] Arif, A., Wang, H., Castaneda, H., and Wang, Y., "Finite-Time Tracking of Moving Platform with Single Camera for Quadrotor Autonomous Landing," IEEE Transactions on Circuits and Systems I: Regular Papers, Vol. 70, No. 6, 2023, pp. 2573–2586. https://doi.org/10.1109/TCSI.2023.3256063.

[6] Yang, L., Wang, X., Liu, Z., and Shen, L., "Robust Online Predictive Visual Servoing for Autonomous Landing of a Rotor UAV," IEEE Transactions on Intelligent Vehicles, Vol. 9, No. 2, 2024, pp. 1–17. https://doi.org/10.1109/TIV.2024.3385283.

[7] Velasco-Sánchez, E., Recalde, L. F., Guevara, B. S., Varela-Aldás, J., Candelas, F. A., Puente, S. T., and Gandolfo, D. C., "Visual Servoing NMPC Applied to UAVs for Photovoltaic Array Inspection," IEEE Robotics and Automation Letters, Vol. 9, No. 3, 2024, pp. 2766–2773. https://doi.org/10.1109/LRA.2024.3360876.

[8] Lee, B., Saj, V., Kalathil, D., and Benedict, M., "Intelligent Vision-based Autonomous Ship Landing of VTOL UAVs," Journal of the American Helicopter Society, Vol. 68, No. 2, 2023, pp. 113–126. https://doi.org/10.4050/JAHS.68.022010.

[9] Gupta, P. M., Pairet, E., Nascimento, T., and Saska, M., "Landing a UAV in Harsh Winds and Turbulent Open Waters," IEEE Robotics and Automation Letters, Vol. 8, No. 2, 2022, pp. 744–751. https://doi.org/10.1109/LRA.2022.3231831.

[10] Hendrick, C. M., Jaques, E. R., Horn, J. F., Langelaan, J. W., and Sydney, A. J., "Model-Scale Evaluation of Autonomous Ship Landing Guidance and Control Modes," Journal of the American Helicopter Society, Vol. 69, No. 2, 2024, pp. 1–20. https://doi.org/10.4050/JAHS.69.022008.

[11] Greer, W. B., and Sultan, C., "Shrinking horizon model predictive control method for helicopter-ship touchdown," Journal of Guidance, Control, and Dynamics, Vol. 43, No. 5, 2020, pp. 884–900. https://doi.org/10.2514/1.G004374.

[12] de Paula, L. G. L., Dwivedi, V. S., Shin, H.-S., and Tsourdos, A., "Towards Autonomous Landing on Dynamic Naval Surfaces Under Harsh Sea State Motion," AIAA SCITECH 2025 Forum, Orlando, Florida, USA, 2025, pp. 1–23. https://doi.org/10.2514/6.2025-2607.

[13] Kang, Y., Park, B. J., Cho, A., Yoo, C. S., Kim, Y., Choi, S., Koo, S. O., and Oh, S., "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, Vol. 19, No. 4, 2018, pp. 994–1005. https://doi.org/10.1007/s42405-018-0081-8.

[14] Recalde, L. F., Varela, J., Guevara, B. S., Andaluz, V., Gimenez, J., and Gandolfo, D., "Constrained Visual Servoing of Quadrotors Based on Model Predictive Control," IFAC-PapersOnLine, Vol. 55, No. 37, 2022, pp. 353–360. https://doi.org/10.1016/j.ifacol.2022.11.209, 2nd Modeling, Estimation and Control Conference MECC 2022.

[15] Lee, D., Lim, H., Kim, H. J., Kim, Y., and Seong, K. J., "Adaptive Image-Based Visual Servoing for an Underactuated Quadrotor System," Journal of Guidance, Control, and Dynamics, Vol. 35, No. 4, 2012, pp. 1335–1353. https://doi.org/10.2514/1.52169.

[16] Jabbari Asl, H., Oriolo, G., and Bolandi, H., "An adaptive scheme for image-based visual servoing of an underactuated UAV," International Journal of Robotics and Automation, Vol. 29, No. 1, 2014, pp. 1–13. https://doi.org/10.2316/Journal.206.2014.1.206-3942.

[17] Applebee, T., and Baitis, A., "Response Amplitude Operator Predictions for the USS Belknap (DLG-26) And USS Joseph Hewes (DE-1052) Class Destroyers," Technical Report ADA002233, US Naval Ship Research and Development Center, Bethesda, Maryland, 1974.

[18] Lee, B., "On the Complete Automation of Vertical Flight Aircraft Ship Landing," Ph.D. thesis, Aerospace Engineering, Texas A&M University, College Station, Texas, USA, 2021. URL https://hdl.handle.net/1969.1/195166.

[19] Nelson, J., "Dynamic Interface Testing of Unmanned Aircraft Systems," Proceedings of the NATO Science and Technology Organization Symposium on Flight Testing of Unmanned Aerial Systems (SCI-269), 2015, pp. 16–1–16–14. URL https://api.semanticscholar.org/CorpusID:201917125.

[20] Quintero, J. M. C., Garcia-Nieto, S., Simarro, R., and Ignatyev, D. I., "Improving Racing Drones Flight Analysis: A Data-Driven Approach Using Motion Capture Systems," Drones, Vol. 8, No. 12, 2024, p. 742. https://doi.org/10.3390/drones8120742.

[21] Wächter, A., and Biegler, L. T., "On the Implementation of a Primal-Dual Interior Point Filter Line Search Algorithm for Large-Scale Nonlinear Programming," Mathematical Programming, Vol. 106, No. 1, 2006, pp. 25–57. https://doi.org/10.1007/s10107-004-0559-y.

[22] de Paula, L. G. L., Dwivedi, V. S., Shin, H.-S., and Tsourdos, A., "Visual Servoing MPC framework for Shipboard Autonomous Landing," AIAA Aviation 2025 Forum, Las Vegas, California, USA, 2025, pp. 1–16. https://doi.org/10.2514/6.2025-3547.

[23] Truong, K. V., Gong, A., Berger, T., Tischler, M. B., and Ivler, C. M., "Development of a Reconfigurable Control Equivalent Turbulence Input Model for Multirotor UAS," Proceedings of the Vertical Flight Society 77th Annual Forum & Technology Display, Vertical Flight Society, Virtual, 2021, pp. 1–9. https://doi.org/10.4050/F-0077-2021-16833.

DSPACE

https://dspace.org/

Visual servoing model predictive control for autonomous shipboard rotorcraft landing in high sea states

de Paula, Luís G. L.; Quintero, Jose M. Castiblanco; Shin, Hyo-Sang; Tsourdos, Antonios

2026-12-31
American Institute of Aeronautics and Astronautics (AIAA)
https://doi.org/10.2514/1.g009512
de Paula LGL, Quintero JMC, Shin H-S, Tsourdos A. (2026) Visual servoing model predictive control for autonomous shipboard rotorcraft landing in high sea states. Journal of Guidance, Control, and Dynamics, Available online 19th January 2026

Downloaded from a DSpace repository. https://dspace.org/