2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Robust Precision Landing of a Quadrotor with Online Temporal Scaling Adaptation of Dynamic Movement Primitives

Kongkiat Rothomphiwat, Prakarn Jaroonsorn, Pakpoom Kriengkomol, Poramate Manoonpong*

***Abstract*—In this work, we address the challenges of robust precision landing maneuvers for a quadrotor on both stationary and moving ground targets in the presence of disturbances that can cause the quadrotor to deviate from its desired trajectory, leading to maneuver failure. To overcome this, we propose a novel online adaptive trajectory planning approach based on the online temporal scaling adaptation of dynamic movement primitives (DMPs). This adaptation enables the desired trajectory to be dynamically adjusted in response to tracking errors and the goal's state. Consequently, our proposed approach enhances accuracy, precision, and safety during landing maneuvers. The effectiveness of the approach is evaluated through comprehensive experiments conducted in both physical simulations and real-world environments, covering various disturbance scenarios.**

***Index Terms*—Drones, Adaptive trajectory planning, Dynamic movement primitives, Precision landing, Robust maneuver, Moving target**

I. Introduction

Quadrotors are becoming increasingly important in various applications, such as search and rescue [1], [2], inspection [3], and package delivery [4], [5], due to their agility and maneuverability. Precision landing is a critical aspect of many quadrotor applications since it enables the delivery of payloads with high accuracy and reliability, reduces the risk of damage to the quadrotor and the environment, and improves the effectiveness of tasks such as search and rescue. However, the precision landing of a quadrotor is a challenging issue that must be addressed to make these applications more reliable and efficient.

One of the main challenges of quadrotor precision landing is handling disturbances in the environment, such as wind gusts, which can affect the quadrotor's trajectory and stability [6]. These disturbances can cause the quadrotor to deviate from its desired path, leading to inaccurate or unsafe landing.

Kongkiat Rothomphiwat is with the Bio-inspired Robotics and Neural Engineering Laboratory, School of Information Science and Technology, Vidyasirimedhi Institute of Science and Technology, Rayong, Thailand (e-mail: kongkiat.r_s24@vistec.ac.th)

Prakarn Jaroonsorn and Pakpoom Kriengkomol are with AI and Robotics Ventures Company Limited (ARV), Bangkok, Thailand (email: prakarnj@arv.co.th and pakpoomk@arv.co.th)

Poramate Manoonpong is with the Bio-inspired Robotics and Neural Engineering Laboratory, School of Information Science and Technology, Vidyasirimedhi Institute of Science and Technology (VISTEC), Rayong, Thailand, and also the Embodied Artificial Intelligence and Neurorobotics Laboratory, SDU Biorobotics, the Mærsk Mc-Kinney Møller Institute, University of Southern Denmark, Odense, Denmark, Odense, Denmark (e-mail: poramate.m@vistec.ac.th*)(corresponding author)

Digital Object Identifier (DOI): see the top of this page.

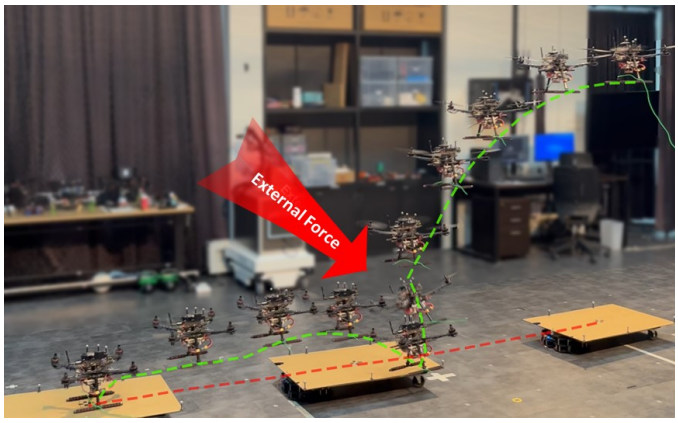


Fig. 1: An autonomous quadrotor controlled by our proposed method performs a landing maneuver on a moving platform while also dealing with external force (indicated by a red arrow) randomly produced by a demonstrator.

Additionally, in many real-world scenarios, the target landing site is not always stationary and may involve a moving vehicle (e.g., ground vehicle [7]). Landing on a moving target adds another level of complexity to the problem since the quadrotor must adjust its speed and trajectory in real-time to land accurately and safely.

Several methods have been proposed to address the challenge of quadrotor precision landing. In [8], a system is proposed that uses a trajectory planning algorithm to generate an optimal feasible trajectory for landing on a moving platform. The algorithm is designed to be computationally efficient, enabling the system to re-plan the trajectory quickly and frequently to adapt to environmental changes. Another technique proposed by [9] is a vision-based guidance technique with a log polynomial closing velocity controller for autonomous landing on a moving target. The technique is based on a pure pursuit guidance law that adjusts the quadrotor's velocity in real-time to converge to the landing target, even if the target moves or the quadrotor experiences disturbances. Other works, involving a disturbance observer-based controller (DOB) [7] and a backstepping position controller with an adaptive rule [10], focus on the controller design to reduce the influence of external disturbances during landing on a moving target. However, most of these works focus solely on disturbances that exhibit repetitive or predictable patterns with low frequency and magnitude and might not effectively manage excessively large or unpredictable disturbances, potentially leading to unsafe maneuvers.

Some works use learning-based methods for quadrotor tasks like precision landing and perturbation handling [11], [12], [13], but these methods require extensive data collection and training. Additionally, unexpected conditions during operation add complexity, necessitating further training. Meanwhile, recent advancements, like Time Delay Estimation (TDE), address state-dependent uncertainties but often rely on conservative assumptions that may not fully capture their complexity [14].

From this point of view, in this paper, we propose a novel online adaptive trajectory planning approach that leverages the principles of dynamic movement primitives (DMPs) as established in [15]- [16], while integrating the most effective aspects of these works to safely and accurately handle both predictable and unpredictable disturbances during landing maneuvers on stationary and moving platforms. We demonstrate the effectiveness of our approach through simulations and real-world experiments. In summary, this work provides the following significant key contributions:

- A new approach is proposed for the robust precision landing of an autonomous quadrotor for landing on stationary and moving targets while handling predictable and unpredictable disturbances in both simulation and the real world.
- A novel online temporal scaling mechanism for DMPs is introduced, aimed at reducing real-time error tracking and simultaneously handling moving targets. This mechanism has not been proposed by others [15]- [16].

II. Preliminaries

A. Dynamic Movement Primitives (DMPs)

DMPs for encoding complex trajectories using motion primitives were first proposed in [15] and described in detail in [17], subsequently emerging as promising techniques for generating robust and adaptive motion trajectories of a time-independent system. DMPs are widely used in robotics and effectively handle uncertainties and disturbances. They can represent both discrete and rhythmic motion, adjustable in space and time while preserving overall trajectory shape. This enables reactive planning via sensory feedback, ensuring robustness in unpredictable environments with wind, obstacles, and disturbances.

The classical DMP formulation in [15] and [17] for the discrete DMP consists of the *transformation system* shown in (1) and (2) which encodes the trajectory in n degrees of freedom (DoFs). All degrees of freedom are synchronized with the *canonical system* (3) which controls the evolution of the phase variable:

$$\tau \dot{z} = \alpha_y(\beta_y(g - y) - z) + diag(g - y_0)f(s), \tag{1}$$

$$\tau \dot{y} = z, \tag{2}$$

$$\tau \dot{s} = -\alpha_x s, \tag{3}$$

where $y \in \mathbb{R}^n$ is the position initialized at y_0. $z \in \mathbb{R}^n$ is the scaled velocity. $g \in \mathbb{R}^n$ is the goal position. $diag(g - y_0) \in \mathbb{R}^{n \times n}$ is a diagonal matrix which controls the movement amplitude. $s \in \mathbb{R}$ is a phase variable with an initial value equal to 1. $\tau \in \mathbb{R}$ is a temporal scaling term, normally equal to the duration of the desired trajectory T. α_y and β_y are set as constant parameters by $\alpha_y = 4\beta_y$ determining the dynamics of the critically damped system. α_x is chosen under the condition that $s \approx 0$ at $y = g$. The forcing term $f(s) \in \mathbb{R}^n$ determining the shape of the trajectory uses a weighted combination of Gaussian basis functions $\Psi_i(s)$ as follows:

$$f(s) = \frac{\sum_{i=1}^{N} w_i \Psi_i(s)}{\sum_{i=1}^{N} \Psi_i(s)} s, \tag{4}$$

$$\Psi_i(s) = \exp\big(-h_i(s - c_i)^2\big), \tag{5}$$

where $w_i \in \mathbb{R}^n$ are the weights of N Gaussian basis functions. c_i are the centers of the kernel distributed along the phase of the trajectory. h_i are the widths of the kernel. The weights (w_i) are chosen using imitation learning, as described in [15], [17] through the use of locally-weighted linear regression (LWR).

B. Dynamic Movement Primitives for Recovering from Disturbances

In the case of disturbances, the approach first proposed in [17] for recovering from disturbances and was improved in [16] by suggesting the coupling term C_t and adapting the temporal scaling term in real-time as follows:

$$e_t = (1 - \alpha_e)(y_s - y) - \alpha_e e_{t-1}, \tag{6}$$

$$C_t = k_t e_t, \tag{7}$$

$$\tau_a = \tau + k_c e_t^2, \tag{8}$$

where α_e, k_t, and k_c are constant gain parameters. Equation (6) is a low-pass filter of the tracking error $(y_s - y)$ and $y_s \in \mathbb{R}^n$ is the system's position. τ_a is an adaptive temporal scaling term affecting the time constant of both transformation and canonical systems. The significant tracking error slows down the evolution of the phase variables s that control the state y to prevent it from deviating too far from system state y_s. C_t in (7) is a spatial coupling term added to (1), forcing y to converge to y_s:

$$\tau_a \dot{z} = \alpha_y(\beta_y(g - y) - z) + diag(g - y_0)f(s) + C_t, \tag{9}$$

$$\tau_a \dot{y} = z, \tag{10}$$

$$\tau_a \dot{s} = -\alpha_x s. \tag{11}$$

C. Dynamic Movement Primitives for Moving Goals

The classical DMP can adapt to a new goal but only works for stationary goals. Thus, [18] and [19] proposed the modification of the classical DMP for moving goals by modifying the transformation system (1) under the assumption that the system knows the goal position and velocity online as follows:

$$\tau \dot{z} = (1 - s)\alpha_y(\beta_y(g - y) + \tau(\dot{g} - \dot{y})) + \Omega, \tag{12}$$

$$\Omega = diag(g - y_0)f(s), \tag{13}$$

$$\tau \dot{y} = z. \tag{14}$$

The weight selection process and other formulations in this modified DMP framework are identical to the classical DMP formulation. The additional term $(1 - s)$ is proposed by [18]. Its purpose is to prevent sudden changes in the acceleration

target at the beginning of the movement, thus reducing the potential for system damage.

The additional system (proposed by [19]) augments the DMP system by introducing a temporal scaling adaptation law (15). This ensures the system maintains the demonstrated trajectory pattern while reaching a moving goal and avoids generating high-velocity commands that exceed the robot's capacity.

$$\dot{\tau} = -\kappa_\tau(\tau - \tau_g) + \dot{\tau_g}, \tag{15}$$

$$\tau_g = \frac{\|g - y_0\|}{\|g_d - y_{0,d}\|}\tau_d. \tag{16}$$

κ_τ is a positive gain parameter controlling the adaptive rate of the temporal scaling term. $g_d, y_{0,d}$, and τ_d represent the goal position, initial position, and duration of the demonstrated trajectory, respectively. To ensure convergence and reaching of the moving goal, τ_g must be bounded by the maximum and minimum distance between $g(t)$ and y_0.

D. Limitations of Existing DMPs

Existing DMPs have limitations in handling unexpected disturbances and moving goals concurrently. The classical DMP (II-A) performs well with stationary goals but struggles with disturbances and moving targets. The disturbance compensation-based DMP (II-B) handles disturbances but cannot adapt to moving goals. The moving goal handling-based DMP (II-C) scales the trajectory to a moving goal but lacks disturbance handling. Therefore, a new adaptive trajectory planning method is needed to address this issue by combining the strengths of each DMP. Thus, this study proposes an adaptive trajectory planning approach based on DMP with online temporal scaling adaptation (DMP-OTSA) that can effectively handle disturbances and moving goals simultaneously.

III. System Overview

As shown in Fig.2, our quadrotor control system consists of two primary components: the proposed adaptive trajectory planning system (DMP-OTSA, a high-level control system) and a low-level control system. This section provides a detailed explanation and description of these components.

A. High-level Control for Robust Precision Landing on Stationary and Moving Goals

Our high-level system is responsible for generating the real-time desired position, velocity, acceleration, and yaw commands of a quadrotor. To achieve a robust precision landing maneuver, an adaptive high-level motion planner is required to generalize both stationary and moving goals while being robust against disturbances. We utilize DMPs to solve the task due to their ability to scale and adapt to goals and unexpected conditions online. We combine the advantage of each DMP, described in Sections II-B and II-C by:

i) using the transformation system and the canonical system to acquire the ability to scale the prescribed trajectory to follow a moving goal, as the following term:

$$\tau\dot{z} = (1-s)\alpha_y(\beta_y(g-y) + \tau(\dot{g} - \dot{y})) + \Omega, \tag{17}$$

$$\Omega = \text{diag}(g - y_0)f(s), \tag{18}$$

$$\tau\dot{y} = z, \tag{19}$$

$$\tau\dot{s} = -\alpha_x s, \tag{20}$$

ii) learning the forcing term based on the LWR to imitate the prescribed trajectory as explained in Section II-A,

iii) using the temporal scaling adaptation law from (15), and modifying τ_g from (16), resulting in the following terms:

$$\dot{\tau} = -\kappa_\tau(\tau - \tau_g) + \dot{\tau_g}, \tag{21}$$

$$\tau_g = \frac{\|g - y_0\|}{\|g_d - y_{0,d}\|}\tau_d + k_c e_t^2, \tag{22}$$

$$e_t = (1-\alpha_e)\big|y_s - y\big| - \alpha_e e_{t-1}. \tag{23}$$

The online temporal scaling adaptation law (22) is proposed here for the first time. This adaptation law addresses two factors: time scaling by distance (first term) from the goal g to the start point y_0, and tracking errors (second term) resulting from disturbances. These scaling terms are utilized to prevent the generation of high-velocity commands when following the goal and effectively respond to disturbances that can lead to significant tracking errors. As a result, a quadrotor is able to recover from disturbances and seamlessly continue along the desired trajectory. The low-pass filter of the absolute value of tracking error in (23) mitigates the negative effects of disturbances that can cause frequent and rapid sign changes in the tracking error. This improves the ability of the system to preserve the significant tracking error state, facilitating recovery from the position error. Due to space limitations, further stability analysis of the control system can be found at https://r-kongkiat.github.io/DMP_OTSA/.

B. Prescribed Landing Trajectory

To ensure a smooth and stable landing for a quadrotor, the reference landing trajectory to be imitated by the DMP system is designed to allow the path in the z-axis to start and converge to the final position after the paths in the x and y axes, using the minimum jerk trajectory for a smooth and continuous trajectory that starts and ends at zero velocity and zero acceleration. The trajectory is given by the following equations:

$$x(t) = \begin{cases} x_i + (x_f - x_i)\cdot q(\frac{t}{T_m}) & \text{if } 0 \le t \le T_m \\ x_f & \text{if } T_m < t \le T \end{cases}, \tag{24}$$

$$y(t) = \begin{cases} y_i + (y_f - y_i)\cdot q(\frac{t}{T_m}) & \text{if } 0 \le t \le T_m \\ y_f & \text{if } T_m < t \le T \end{cases}, \tag{25}$$

$$z(t) = \begin{cases} z_i & \text{if } 0 \le t < T_s \\ z_i + (z_f - z_i)\cdot q(\frac{t-T_s}{T-T_s}) & \text{if } T_s \le t \le T \end{cases}, \tag{26}$$

where:
x_i, y_i, z_i: the initial position,
x_f, y_f, z_f: the final position,
T_m: the intermediate time at which the x and y axes converge,
T_s: the start time for the z-axis,

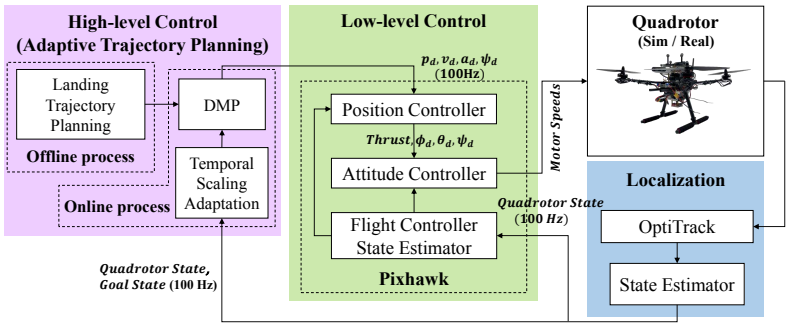


Fig. 2: The diagram shows an overview of our system. The high-level control system (purple box) utilizes the online temporal scaling adaptation of DMP to generate the desired state command. This command is then passed to the low-level control system (green box) responsible for computing motor speeds to the quadrotor (white box). In real system implementation, both the high-level and low-level controllers receive feedback on the quadrotor and goal states from the motion capture system (OptiTrack, blue box).

T: the total time for the trajectory ($T > T_m > T_s$),
$q(u)$: a function that generates the minimum jerk trajectory, defined as:

$$q(u) = 10u^3 - 15u^4 + 6u^5. \tag{27}$$

Note that T_m must be greater than T_s to ensure the trajectory's smoothness; otherwise, the generated path will lack continuity. T_m is crucial for generating a trajectory that combines horizontal tracking and vertical descent simultaneously. This results in a more time-efficient path compared to a trajectory that first achieves horizontal tracking and then descends vertically (which can be represented by the $T_s > T_m$ case). A recommended approach for selecting T_m, T_s, and T is presented in Section IV-B.

C. Low-level Controller

The low-level controller runs on ArduPilot firmware [20] and consists of position and attitude sub-controllers. It receives the desired trajectories from high-level control and computes the desired collective thrust and torque before finally converting it into the motor speed command to the rotors.

IV. Physical Simulation

In this section, we demonstrate the generality and robustness of our proposed trajectory planning approach described in Section III-A by testing it in multiple situations. First, we evaluate the quadrotor's robustness when landing on a stationary goal in Section IV-B and the generality when landing on a moving goal in Section IV-C.

A. Simulation Setup

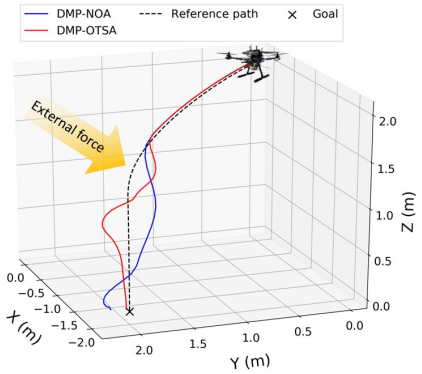
Our algorithm is first implemented with the ArduPilot SITL as a low-level flight controller for a simulated quadrotor in Gazebo. It communicates with the low-level controller via a supported bridge between the Robot Operating System (ROS) and the MAVLink protocol (MAVROS).

B. Stationary Goal

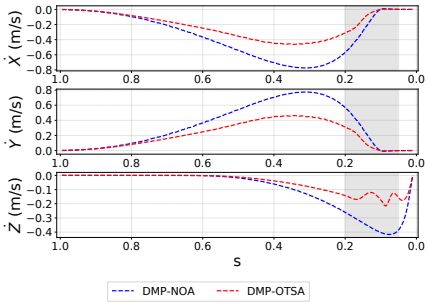
In this experiment, we evaluate the robustness of our proposed approach when the quadrotor lands on a stationary goal. We train the DMP using a given trajectory as described in Section III-B. A prescribed landing trajectory can be obtained for the DMP through imitation learning by imitating a landing trajectory that starts at the initial position of $y_{0,d} = [0.0, 0.0, 0.0]^T$ and ends at the final position of $g_d = [1.0, 1.0, 1.0]^T$. The total duration of the path ($T = \tau_d = 4.83$), is estimated by $\frac{\|g_d - y_{0,d}\|}{V_{max}}$. The x and y axes converge to the final position at $T_m = \tau_d/2$, while the z-axis starts at $T_s = \tau_d/10$. The parameters utilized in our DMP setup are outlined in Table I, with particular attention given to the low-pass parameter α_e. This parameter is empirically adjusted to ensure that minor tracking errors do not significantly impact the overall flight duration, and the trajectory is determined in 3D Cartesian coordinates. The initial duration of the trajectory is determined by $\tau = \frac{\|g - y_0\|}{\|g_d - y_{0,d}\|}\tau_d$.

To test our system, we allow the drone to follow the landing trajectory provided by the DMP at 100 Hz, starting to hover from $y_0 = [0.0, 0.0, 2.0]^T$, and landing on the stationary goal at $g = [-2.0, 2.0, 0.0]^T$. The goal position and velocity are updated at every iteration of the DMP system. During the landing maneuver, we utilize the phase variable s to trigger the application of an external force as a disturbance to the quadrotor at $s = 0.2$ and terminate the landing maneuver at $s = 0.01$. The external force is an impulsive one with $F = [-1.0, 1.0, 0.0]^T$ applied to the quadrotor for 3 s.

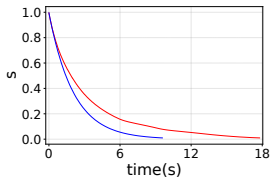
We compare our adaptive trajectory planning method (DMP-OTSA) with the pre-planning trajectory, which is an offline process with fixed movement duration and lacks online adaptability; the pre-planning trajectory can be interpreted as DMP without temporal error adaptation (described as DMP-NOA). Figure 3 illustrates a comparison of the performance of DMP-OTSA (red line) and DMP-NOA (blue line). When the quadrotor experiences disturbances (highlighted in the gray region and the yellow arrow), DMP-OTSA automatically adjusts the τ value, associated with the tracking error (Fig. 3d), to slow down the system evolution and velocity command, aiding in quadrotor recovery before continuing along the trajectory. Consequently, this adjustment prolongs the flight duration, as depicted in Fig. 3c, but proves valuable in minimizing the final landing position error. The results in Fig. 3a show that DMP-NOA fails to compensate for disturbances, resulting in a larger end-of-maneuver error (17.55 cm) compared to DMP-OTSA, which achieves better performance with a smaller error (3.48



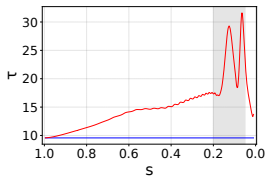
(a) The 3D path of the quadrotor while following the landing trajectory (dashed black line) under the presence of disturbance (indicated by the yellow arrow)



(b) The reference velocities generated by the DMPs



(c) Phase variable



(d) Temporal Scaling factor

Fig. 3: The simulation results for the stationary goal when it was affected by disturbances (highlighted in the gray region) for the cases: the DMP-NOA (blue line) and the DMP-OTSA (red line).

cm).

Additionally, we also evaluate the robustness of our approach when landing on a stationary goal with wind disturbance. The setup is the same as for the previous experiment, but the disturbances are applied throughout the flight. Detail of the wind disturbances are provided in Appendix A. The performance of each approach is compared with varying wind speeds (v_{mag}). To select the wind speeds, the wind speed is selected according to the average force output per weight of the quadrotor ($\frac{F}{mg}$) which should range from 0 to 1 in increments of 0.2. In addition, the wind direction (ϕ_{mean}) is randomly set in each test. The parameters used for wind simulation are also shown in Table I. Both approaches are repeated five times for each specified wind speed. Fig.4 shows the comparative results of the experiment. The DMP-OTSA can keep the lowest precision through varying wind speed or force.

C. Moving Goal

In this experiment, we evaluate the robustness and generalization of our approach by transitioning from a stationary to

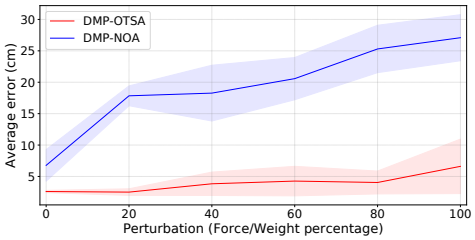


Fig. 4: The simulation results from the robustness test on wind disturbance, varying wind speed, display the average error at touchdown and precision of the quadrotor from five repeated experiments. The DMP-NOA method is indicated by a blue line, and the DMP-OTSA is represented by a red line.

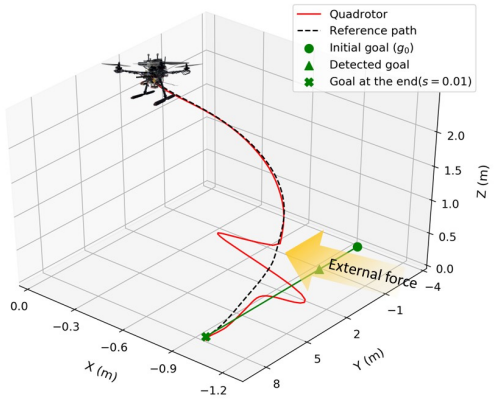
TABLE I: The DMP system (left) and the wind simulation parameters (right)

Parameters	Value
α_y	25
β_y	6.25
α_x	4.6052
N	300
κ_τ	10
k_c	150
α_e	0.98
$\tau_{g,max}$	8.0
$\tau_{g,min}$	1.732

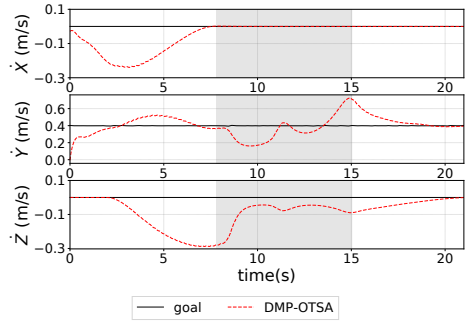
Parameters	Value
m	1.5 kg
C_f	1
A_v	0.1
A_ϕ	30 degree
T	30 s

moving goal. The DMP system is trained and parameterized in the same manner as described in Section IV-B. The moving goal position is initialized at $g_0 = [-1.0, -3.0, 0.0]^T$, moving at a constant velocity of $\dot{g} = [0, 0.4, 0]^T$. The quadrotor begins by hovering at $y_0 = [0.0, 0.0, 2.0]^T$. As soon as the goal approaches (in this case, we provide external information to trigger when the goal reaches $g = [-1.0, 0.0, 0.0]^T$), the DMP is executed to provide a landing trajectory at a frequency of 100 Hz for the quadrotor low-level controller from y_0 to g. The goal position and velocity are also updated in every iteration of the DMP. During the landing maneuver, a disturbance is applied to the quadrotor at $s = 0.05$. The disturbance is also applied as an impulsive force with $F = [2.0, 0.0, 0.0]^T$ for 2 s. The landing maneuver is terminated at $s = 0.01$.

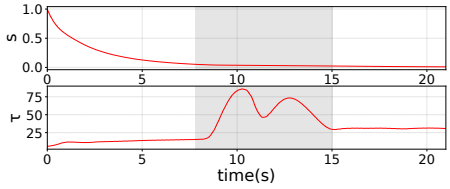
The experiment demonstrates that the DMP-OTSA enables the quadrotor to successfully recover from disturbances and reach a moving target, as depicted in Fig. 5a. This success is due to our adaptation of the temporal scaling term in (22), considering both the tracking error and the relative distance from the goal g to the initial position y_0 simultaneously. Fig. 5c illustrates the modification of the temporal scaling term when the quadrotor encounters a disturbance (highlighted in the gray region) and the moving target throughout the maneuver. This leads to the adaptive velocity command shown in Fig. 5b, which slows down in the case of disturbance and finally converges to the target position and velocity. In contrast, the conventional DMP-NOA [19] depicted in Fig. 6a only takes into account the relative distance, generating the trajectory command to converge to the target's position and velocity, as can be observed in Figs. 6b and 6c. However, since the tracking error is not considered, this approach results in the quadrotor failing to achieve the maneuver and missing the target.



(a) 3D path of the quadrotor (red line) while following the reference trajectory (dashed black line) toward a moving goal under the presence of disturbance (indicated by yellow arrow).

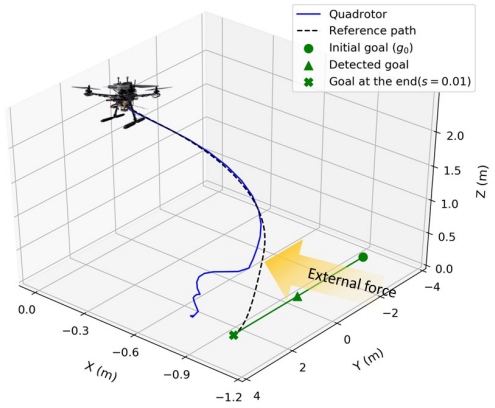


(b) The reference velocity generated by the DMP-OTSA and the goal velocity.

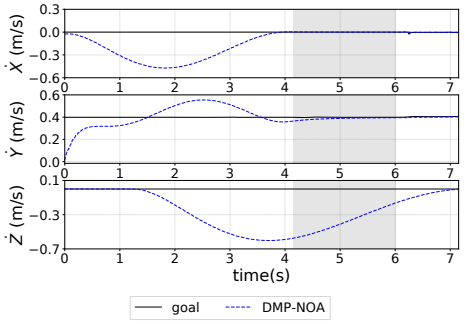


(c) Phase variable and temporal scaling factor.

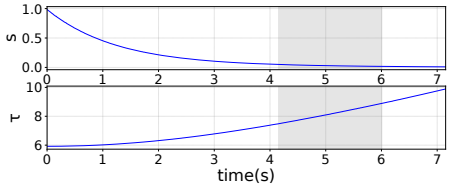
Fig. 5: The simulation results for the moving goal when affected by disturbance (highlighted in the gray region) for the DMP-OTSA.



(a) 3D path of the quadrotor (blue line) while following the reference trajectory (dashed black line) toward a moving goal under the presence of disturbance (indicated by yellow arrow).



(b) The reference velocity generated by the DMP-NOA and the goal velocity.



(c) Phase variable and temporal scaling factor.

Fig. 6: The simulation results for the moving goal when affected by disturbance (highlighted in the gray region) for the DMP-NOA.

V. Real-World Evaluation

This section evaluates the proposed approach in real-world conditions by performing the landing maneuver under disturbances such as a windy environment and external force for both stationary and moving goal cases.

A. Quadrotor Platform

An autonomous drone is developed in this work (depicted in Fig.7). Based on the 500 mm PCB multi-rotor air frame (S500). The key components of the drone consist of 1) Pixhawk (low-level flight controller) running ArduPilot (an open-source low-level flight controller), 2) an onboard computer (Odroid XU4) that utilizes ROS for the implementation of high-level control and communication with both Pixhawk and a motion capture system (OptiTrack). The interface between the onboard computer and the flight controller is established through MAVROS, which is the same protocol mentioned in Section IV-A.

B. Moving Landing Platform

TurtleBot3 serves as a vehicle for dragging a cart, with an acrylic pad (60×90 cm) performing as a moving landing platform (Fig.7). The maximum speed of the platform is 0.2 m/s. During the experiments, the moving platform (TurtleBot3) is controlled manually with a joystick.

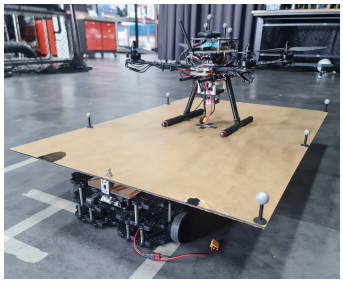
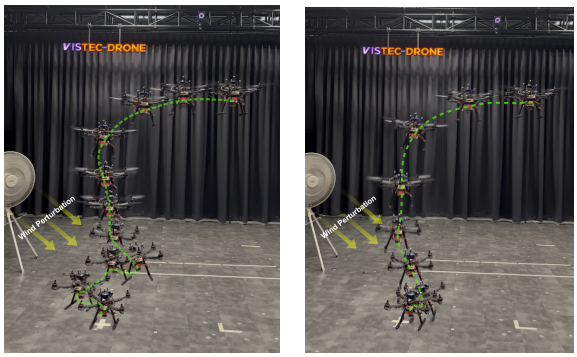


Fig. 7: The quadrotor platform and the TurtleBot-based moving landing platform for real-world experiments.

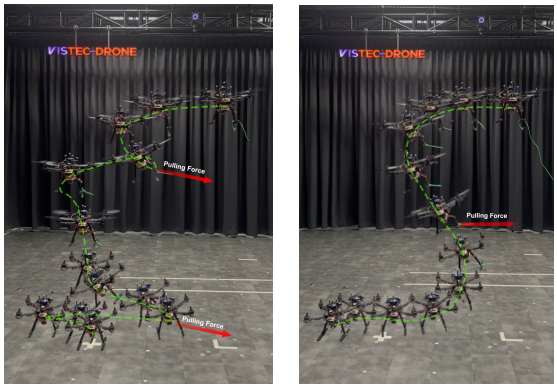
C. Indoor Localization

The motion capture system (9 x motion capture OptiTrack Prime 17W) is utilized to provide the poses of the quadrotor and the moving landing platform at a frequency of 100 Hz. The quadrotor pose is directly passed to the low-level controller state estimator unit. To estimate the state of the moving landing platform (position and velocity), a linear Kalman Filter [21] is employed to filter and smooth the noise in the measurements, using a constant velocity model.



(a) DMP-OTSA (b) DMP-NOA

Fig. 8: The real evaluation of the quadrotor under wind disturbance (indicated by yellow arrows).



(a) DMP-OTSA (b) DMP-NOA

Fig. 9: The real evaluation conducted on the quadrotor with impulsive force disturbances (indicated by red arrows) for two cases. The disturbances are generated by a demonstrator who randomly pulls a rope to produce the force.

D. Stationary Goal

For the stationary goal, two different disturbance conditions (wind and impulsive force) are used to evaluate the DMP-OTSA and the DMP-NOA, with the DMP system being trained and parameterized in the same way as described in Section IV-B. The experiments begin by hovering the quadrotor at $y_0 = [0.0, 0.0, 2.0]^T$, then following the path provided by the DMP at 100 Hz to land on the stationary goal at $g = [0, 1.05, 0]^T$, updating the goal position and velocity in every iteration to the DMP. The maneuver is also terminated at $s = 0.01$.

The results of the quadrotor trajectory for the DMP-OTSA and the DMP-NOA are shown in Figs. 8 and 9. Figure 8 shows the case of the wind disturbance produced by a fan with wind speeds ranging from 1.8 to 2.3 m/s, measured by a digital anemometer and Figure 9 shows the case of the impulsive force disturbance, with a rope being attached to the quadrotor and randomly pulled by a demonstrator. The impulse pulling force is in a range of $15 - 25$ N for 2 s (measured by a digital force gauge attached to the rope). The results show that the proposed DMP-OTSA can deal with the disturbances better than the DMP-NOA. In the experimental evaluation of the wind case, a small deviation of 3.09 cm is produced using our approach, with the DMP-NOA exhibiting a large deviation of 11.62 cm. Similarly, in the impulsive force case, a small deviation of 1.66 cm is achieved with our method, while the DMP-NOA exhibits a large deviation of 42.05 cm.

E. Moving Goal

To demonstrate the moving goal case, we test the DMP-OTSA on the quadrotor to land on the landing platform with and without disturbances. The DMP-OTSA is used to generate the following trajectory to the quadrotor, relying on the position and velocity feedback to the DMP and the quadrotor. The DMP system is trained and parameterized in the same way as described in Section IV-B. The landing platform moves in a straight line at a maximum velocity of 0.2 m/s. The quadrotor starts by hovering at $y_0 = [1.5, 1.0, 2.0]^T$, waiting for the landing platform to move closer, and then starts to follow the path generated from the DMP at 100 Hz to land on the landing platform at $s = 0.01$. In this experiment, the rope is fastened to the quadrotor and randomly pulled by a demonstrator to generate an impulse force disturbance, as described in Section V-D.

Figures. 1 and 10 illustrate the trajectory of the quadrotor while landing on the platform with and without disturbances, respectively. The figures depict the ability of the quadrotor to successfully perform the landing maneuver even in the presence of disturbances. A video demonstrating all real experiments can be seen at https://www.manoonpong.com/DMP-OTSA/video.mp4.

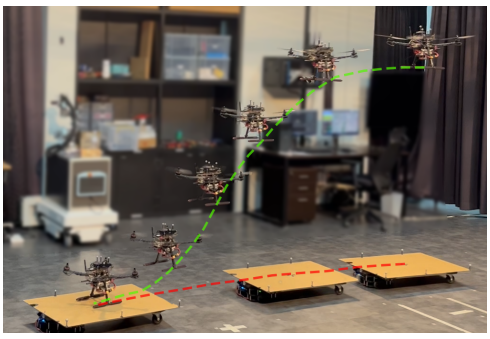


Fig. 10: The real evaluation of the quadrotor driven by the DMP-OTSA when performing a landing maneuver on the moving platform without disturbance.

VI. Discussion And Conclusion

In this study, an online adaptation algorithm for planning the landing trajectory of a quadrotor is presented. The algorithm brings together the best aspects of previously modified

dynamic movement primitives from [18], [19], and [16]. The modified DMP with online temporal scaling adaptation allows the system to adapt the prescribed landing trajectory to moving targets and deal with disturbances by utilizing the current position and velocity of the goal and tracking errors as feedback. The adaptation slows down the evolution of the DMP system, preventing the generation of high-velocity commands and controlling the command within a reasonable range of the quadrotor states. This makes it possible for the quadrotor to perform autonomous landing maneuvers on stationary or moving targets with greater resilience to unexpected disturbances. This approach reduces tracking errors, enhances both accuracy and precision and improves overall safety during the landing procedure, as demonstrated by the results evaluated in physical simulation and real-world conditions.

Various methods, such as those proposed in [8] - [14], have been developed for quadrotors to handle disturbances during landing using trajectory planning and control algorithms in different ways. However, some methods struggle with strong, unpredictable disturbances, which can make the landing unsafe. Furthermore, some approaches, which require extensive data collection and intensive training, or rely on conservative assumptions, add additional workload and become impractical. In contrast, our method eliminates the need for these while successfully enabling precision landing on both static and moving targets under unexpected conditions.

Despite this, our approach has limitations in perception, as it relies solely on external position feedback from a motion capture system and lacks the ability to detect fast-moving targets. Future enhancements will include integrating advanced perception systems, such as depth and/or event-based cameras, to enable fully autonomous flying and landing in real-world outdoor applications.

ACKNOWLEDGMENTS

This work was supported by VISTEC and ARV under the ADAFT project. The authors would like to express their gratitude to Atthanat Harnkhamen for providing assistance during the experiments.

APPENDIX

A. Wind Disturbance

The wind velocity vector v_w is formed by selecting the magnitude of the velocity v_{mag} and direction ϕ_{mean} as follows:

$$v_w(t) = \left(v_{mag} + A_v v_{mag} \sin(\frac{2\pi t}{T})\right) \begin{bmatrix} \cos(\phi) \\ \sin(\phi) \\ 0 \end{bmatrix}, \quad (28)$$

$$\phi(t) = \phi_{mean} + A_\phi \sin(\frac{2\pi t}{T}). \quad (29)$$

The uncertainty of the wind velocity and direction is modeled by introducing the sine wave function in both (28) and (29) at the period T. A_ϕ represents the amplitude of the oscillating direction, while A_v represents the amplitude of the oscillation, which is the percentage of magnitude (v_{mag}).

The force acting on the quadrotor from the effect of the wind is simply modeled similarly to the drag force equation:

$$F = mC_f(v_w - v_{obj}), \quad (30)$$

where m is the mass of an object, C_f is the force coefficient, and v_{obj} is the velocity of an object.

REFERENCES

[1] S. Waharte and N. Trigoni, "Supporting search and rescue operations with uavs," in *2010 International Conference on Emerging Security Technologies*, 2010, pp. 142–147.

[2] E. Z. Mario Silvagni, Andrea Tonoli and M. Chiaberge, "Multipurpose uav for search and rescue operations in mountain avalanche events," *Geomatics, Natural Hazards and Risk*, vol. 8, no. 1, pp. 18–33, 2017.

[3] J. Nikolic, M. Burri, J. Rehder, S. Leutenegger, C. Huerzeler, and R. Siegwart, "A uav system for inspection of industrial facilities," in *2013 IEEE Aerospace Conference*, 2013, pp. 1–8.

[4] D. Mellinger, M. Shomin, N. Michael, and V. Kumar, *Cooperative Grasping and Transport Using Multiple Quadrotors*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 545–558.

[5] K. Feng, W. Li, S. Ge, and F. Pan, "Packages delivery based on marker detection for uavs," in *2020 Chinese Control And Decision Conference (CCDC)*, 2020, pp. 2094–2099.

[6] B. H. Wang, D. B. Wang, Z. A. Ali, B. Ting Ting, and H. Wang, "An overview of various kinds of wind effects on unmanned aerial vehicle," *Measurement and Control*, vol. 52, no. 7-8, pp. 731–739, 2019.

[7] N. Xuan-Mung, N. P. Nguyen, T. Nguyen, D. B. Pham, M. T. Vu, H. L. N. N. Thanh, and S. K. Hong, "Quadcopter precision landing on moving targets via disturbance observer-based controller and autonomous landing planner," *IEEE Access*, vol. 10, pp. 83 580–83 590, 2022.

[8] 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)*, 2017, pp. 200–207.

[9] A. Gautam, M. Singh, P. B. Sujit, and S. Saripalli, "Autonomous quadcopter landing on a moving target," *Sensors*, vol. 22, no. 3, 2022.

[10] 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.

[11] M. O'Connell, G. Shi, X. Shi, K. Azizzadenesheli, A. Anandkumar, Y. Yue, and S.-J. Chung, "Neural-fly enables rapid learning for agile flight in strong winds," *Science Robotics*, vol. 7, no. 66, p. eabm6597, 2022.

[12] L. Bartolomei, Y. Kompis, L. Teixeira, and M. Chli, "Autonomous emergency landing for multicopters using deep reinforcement learning," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 3392–3399.

[13] S. Lee, T. Shim, S. Kim, J. Park, K. Hong, and H. Bang, "Vision-based autonomous landing of a multi-copter unmanned aerial vehicle using reinforcement learning," in *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2018, pp. 108–114.

[14] S. Dantu, R. D. Yadav, S. Roy, J. Lee, and S. Baldi, "Adaptive artificial time delay control for quadrotors under state-dependent unknown dynamics," in *2022 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2022, pp. 1092–1097.

[15] A. Ijspeert, J. Nakanishi, and S. Schaal, "Movement imitation with nonlinear dynamical systems in humanoid robots," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, vol. 2, 2002, pp. 1398–1403 vol.2.

[16] M. Karlsson, F. B. Carlson, A. Robertsson, and R. Johansson, "Two-degree-of-freedom control for trajectory tracking and perturbation recovery during execution of dynamical movement primitives." *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 1923–1930, 2017.

[17] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: Learning attractor models for motor behaviors," *Neural Computation*, vol. 25, no. 2, pp. 328–373, 2013.

[18] J. Kober, K. Mülling, O. Krömer, C. H. Lampert, B. Schölkopf, and J. Peters, "Movement templates for learning of hitting and batting," in *2010 IEEE International Conference on Robotics and Automation*, 2010, pp. 853–858.

[19] L. Koutras and Z. Doulgeri, "Dynamic movement primitives for moving goals with temporal scaling adaptation," in *2020 IEEE International Conference on Robotics and Automation*, 2020, pp. 144–150.

[20] "Arudpilot code overview," 2024, accessed: 2024-08-06. [Online]. Available: https://ardupilot.org/dev/docs/apmcopter-code-overview.html

[21] A. Kelly, "A 3d state space formulation of a navigation kalman filter for autonomous vehicles," 2000.