# Autonomous UAV Landing with Visual Servoing and Semi-Markov Decision Processes

Fabián González-Tejeda, Brenda Ivette García-Maya*, and Hernán Abaunza
Tecnologico de Monterrey, School of Engineering and Sciences



## Abstract

This paper presents an innovative approach to autonomous UAV landing on moving platforms by integrating a Semi-Markov Decision Process (SMDP) framework with a visual servoing control strategy. The proposed method models the UAV's position relative to the landing target using a semi-Markov state space, accounting for varying visibility conditions and the precise positioning required for a successful landing. By incorporating a reward function, the SMDP framework enables the UAV to evaluate trade-offs between different actions and dynamically optimize its trajectory. This approach enhances the UAV's ability to adapt to platform movement, handle uncertainty, and improve landing performance in unpredictable environments. The contribution of this work lies in the novel application of SMDPs combined with visual servoing, offering a robust, adaptable solution to the challenges of autonomous landing in real-world, dynamic conditions.

## 1 Introduction

Unmanned Aerial Vehicles (UAVs) have rapidly advanced in response to the growing demand for real-time applications in dynamic environments. Consequently, autonomous landing systems have become essential for various tasks [1], including package delivery [2], search and rescue [3], and aerial inspections requiring timely decisions [4].

These systems depend on advanced algorithms and sensors to perform precise, reliable landings. Among state-of-the-art methods, visual servoing [5, 6] stands out as a popular approach [7, 8], providing accurate control by processing visual feedback to guide the UAV during descent. However, visual servoing faces significant challenges related to environmental variability, occlusions, sensor inaccuracies, and computational limitations, demanding more sophisticated strategies for robust operation [9, 10].

*Corresponding author
F. González-Tejeda: A01634223@tec.mx, ORCID: 0009-0002-1597-6244,
B.I. García-Maya: brenda.gmaya@tec.mx, ORCID: 0000-0002-9945-3784.
H. Abaunza: habaunza@tec.mx ORCID: 0000-0001-5325-730X.
Tecnologico de Monterrey, School of Engineering and Sciences
Guadalajara Campus, Av. General Ramón Corona 2514, 45201 Zapopan, Jalisco, México.

To address these issues, Kalman filters have been widely adopted to model and manage uncertainty during landing maneuvers [11]. While effective for filtering noise and estimating states, they may struggle under highly unpredictable conditions—particularly when vision-related problems arise, such as lighting changes, occlusions, target motion, and UAV inclination variations, especially without gimbal stabilization.

To improve robustness under uncertainty, alternative probabilistic frameworks like Markov chains have been explored. By modeling the landing process as discrete states with transition probabilities, Markov chains enable detailed analysis of possible outcomes based on UAV actions. Extending this concept, Markov Decision Processes (MDPs) [12] introduce decision-making into the framework, allowing UAVs to handle stochastic environments by associating actions with rewards and transition probabilities [13].

The use of Markov models represents a significant step toward overcoming UAV landing challenges, improving reliability and adaptability in uncertain scenarios [14]. However, standard Markov processes assume sojourn times follow geometric or exponential distributions, limiting their flexibility. Semi-Markov systems address this limitation by allowing any probability distribution for sojourn times [15].

This work aims to improve autonomous UAV landing on moving platforms by integrating a Semi-Markov Decision Process (SMDP) framework with visual servoing. The SMDP models the UAV's state relative to the target, accounting for varying visibility and precise positioning needed for target locking and landing. A reward function guides the UAV to evaluate trade-offs and select actions that maximize landing success probability. This approach enables adaptation to platform motion, uncertainty management, and trajectory optimization. The main contribution is this novel combination of SMDP and visual servoing, offering a robust, adaptable solution for real-world UAV landing scenarios.

The remainder of this paper is organized as follows: Section 2 describes the proposed methodology; Section 3 details the visual servoing control strategy; Section 4 presents the SMDP framework for autonomous decision-making; Section 5 discusses the experimental validation; and Section 6 provides conclusions and future perspectives.

## 2 Methodology

The proposed solution integrates **Visual Servoing** and **Semi-Markov Decision Processes (SMDPs)** to train a reliable UAV landing system, as illustrated in Figure 1. **Visual**

**Servoing** is responsible for state estimation, providing real-time information about the relative position of the landing platform. **SMDPs**, on the other hand, enable decision-making by modeling the system's behavior over time.

The system processes two main inputs:

1. **Real-time platform position (Input 1a):** The UAV's onboard camera captures the relative position of the landing platform. This information is converted into discrete states representing the platform's location in relation to the UAV.

2. **Experimental flight data (Input 1b):** A dataset is collected from previous flights, recording the observed states, actions taken, and corresponding outcomes. This historical data is used to estimate the probabilities of different system behaviors over time, forming a transition probability matrix.

Using this probability matrix, the system evaluates the expected rewards for different actions in each state. It computes the optimal action the UAV should take to maximize its chances of a successful landing. Once computations are completed (Steps 3, 4, 5, and 6 in Figure 1), the drone controller (acting as the pilot) selects the most effective decision for the given scenario.

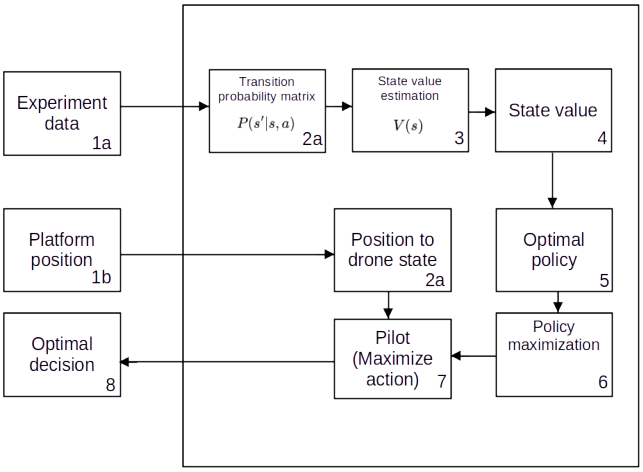



Figure 1: UAV landing decision process algorithm diagram

This structured approach allows the UAV to dynamically adapt its landing strategy based on real-time observations and learned experience, improving accuracy and reliability.

The relative pose of the moving platform is estimated by detecting visual markers (e.g., QR codes) from the camera feed. The visual data is interpreted and converted into a series of discrete states (detailed in Section 4), which are used to ensure real-time awareness of the system's status.

Then, a SMDP, models the problem as a sequence of states, actions, and rewards. The state space captures the UAV's relative position to the target and the status of the landing platform, with terminal states representing either a successful landing or a failed attempt. The action space defines the UAV's possible maneuvers, such as adjusting circular velocity, moving in specific directions, or centering the target based on visual feedback.

State transition probabilities account for system uncertainties, representing the likelihood of moving from one state to another after performing a specific action. The reward function is designed to promote behaviors that guide the UAV toward the target while penalizing inefficient actions.

The objective is to find an optimal policy $\pi^*(s)$, which is a function that maps each state $s$ to an optimal action $a$, maximizing the expected cumulative reward:

$$\pi^*(s) = \arg\max_{\pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right],$$

where $\gamma$ is the discount factor that determines the importance of future rewards. The reward function $R$ is carefully crafted to minimize the landing error and ensure that the UAV makes decisions that lead to a safe and accurate landing.

The optimal policy is computed using linear programming techniques. These algorithms enable the UAV to learn the optimal policy through trial and error, updating its actions based on the observed rewards. The integration of Visual Servoing and SMDPs allows the system to adapt dynamically to the changing environment, making real-time adjustments based on the current state.

The system operates in a feedback loop, where Visual Servoing continuously estimates the platform's position and the SMDP evaluates the state and selects an optimal action. This loop ensures that the UAV can adjust its trajectory in response to platform motion and other uncertainties, improving the likelihood of a successful landing.

## 3 UAV MODELING AND VISUAL SERVOING STRATEGY

The UAV's dynamics are governed by:

$$m\ddot{\mathbf{p}} = \mathbf{R}(\mathbf{q}) \cdot \mathbf{f_b} - m\mathbf{g}, \tag{1}$$

$$\mathbf{J}\dot{\boldsymbol{\omega}} = \boldsymbol{\tau} - \boldsymbol{\omega} \times (\mathbf{J}\boldsymbol{\omega}), \tag{2}$$

where $m \in \mathbb{R}$ is the UAV mass, $\ddot{\mathbf{p}} \in \mathbb{R}^3$ its acceleration (inertial frame), $\mathbf{R}(\mathbf{q}) \in \mathbb{R}^{3\times3}$ the rotation matrix from quaternion $\mathbf{q} \in \mathbb{H}$, $\mathbf{J} \in \mathbb{R}^{3\times3}$ the inertia matrix (body frame), and $\boldsymbol{\omega} = [\omega_x \ \ \omega_y \ \ \omega_z]^T$ the angular velocity. Gravity is given by $\mathbf{g} = [0\,,\, 0\,,\, -9.81]^T$ m/s$^2$.

The internal control system operates as follows:

- **Inputs:** Desired translational velocity $\mathbf{v_d} = [v_{d_x} \ \ v_{d_y} \ \ v_{d_z}]^T$ and yaw angular velocity $\omega_{z_d} \in \mathbb{R}$.
- **Outputs:** Thrust force $f_b \in \mathbb{R}$ and control torque $\boldsymbol{\tau} \in \mathbb{R}^3$ (body frame).

**Assumption 1** *The internal controller (provided by the manufacturer) ensures that* $\mathbf{v} \rightarrow \mathbf{v_d}$*, and* $\boldsymbol{\omega} \rightarrow [0\,,\,0\,,\,\omega_{z_d}]^T$*, where* $\mathbf{v} = [v_x \;\; v_y \;\; v_z]^T = \dot{\mathbf{p}}$ *is the translational velocity (inertial frame) and* $\boldsymbol{\omega}$ *the angular velocity (body frame).*

### *3.1 Visual Servoing for Autonomous Landing*

The use of Image-Based Visual Servoing (IBVS) in UAV control relies on image feedback to achieve precise positioning [16]. Unlike position-based methods, IBVS operates directly in the image plane, reducing dependency on accurate 3D scene reconstruction. This approach is particularly effective for tasks such as landing, object tracking, and navigation in dynamic environments. The operational scheme of **Image-Based Visual Servoing (IBVS)** for multirotor UAVs is illustrated in Figure 2, highlighting the feedback loop between visual data and control commands. The diagram represents the relationship between the **image plane** and the **drone position plane**. The UAV moves along the $x$, $y$, and $z$ axes, representing the spatial coordinates in which it can maneuver freely.

The **image plane** is a two-dimensional (2D) representation of the captured scene, where positions are defined in pixel coordinates $(u, v)$. The observed image is composed of pixel coordinate sets, with the image center defined as $(u_0, v_0)$. To align the image center with a target, we calculate the distance between the center and any point in the image plane, denoted as $p = [u, v]$.

Once the desired trajectory is determined in the image plane, mapping it to the drone's movement requires a transformation into the **position plane**, where coordinates are expressed as $p = [x, y, z]$. This transition between the image plane and the position plane is achieved using the **Jacobian matrix**, which establishes the mathematical relationship between pixel displacement and UAV motion. More details on this transformation are provided in the following sections.

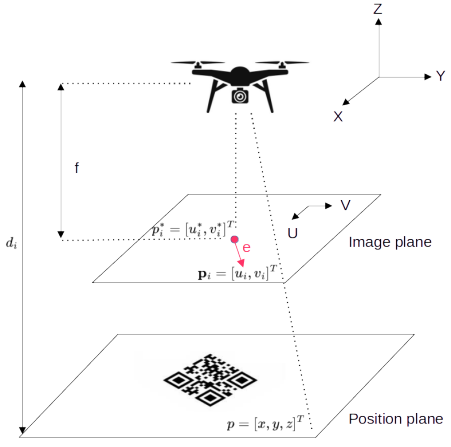



Figure 2: IBVS control scheme for a multirotor UAV, illustrating image plane error correction for precise target alignment.

#### 3.1.1 Error Definition

The error in IBVS is defined as the distance difference between an actual point and our desired position:

$$\mathbf{e} = \mathbf{p}_i - \mathbf{p}_i^* \tag{3}$$

where: $\mathbf{p}_i = [u_i, v_i]^T$ denotes the observed image feature position, and $\mathbf{p}_i^* = [u_i^*, v_i^*]^T$ represents the desired image feature position.

The image position is represented in the image frame, described by the camera position. The transformation with the jacobian matrix help transform the desired position in the camera frame to the a desired velocity for the UAV.

#### 3.1.2 Pixel Velocity Control Law

To understand the pixel velocity of an image it is important to compute the relationship between the pixel velocity and the camera velocity, to achieve this we multiple the camera velocity by the image Jacobian matrix which is given by:

$$\dot{\mathbf{p}}_i = J\boldsymbol{\nu} \tag{4}$$

where:

- $\dot{\mathbf{p}}_i = [\dot{u}, \dot{v}]^T$: pixel velocity.
- $\boldsymbol{\nu} = [\mathbf{v}^T, \boldsymbol{\omega}^T]^T = [v_x, v_y, v_z, \omega_x, \omega_y, \omega_z]^T$: UAV translational and rotational velocities.
- $J$: Jacobian matrix relating UAV velocities to image feature velocities:

$$J = \begin{bmatrix} -\frac{f_x}{d_i} & 0 & \frac{u_i}{d_i} & \frac{u_i v_i}{f_x} & -(f_x + \frac{u_i^2}{f_x}) & v_i \\ 0 & -\frac{f_y}{d_i} & \frac{v_i}{d_i} & \frac{v_i^2}{f_y} & -\frac{u_i v_i}{f_y} & -u_i \end{bmatrix} \tag{5}$$

where $f_x, f_y$ are the focal lengths, $d_i$ is the depth of the feature, and $u_i, v_i$ are the coordinates of the feature in the image plane.

#### 3.1.3 Control Law

Considering Assumption 1, a control law that computes the desired camera velocity can be expressed as:

$$\boldsymbol{\nu}_d := -\lambda J^{\dagger}\mathbf{e} \tag{6}$$

where $\lambda$ denotes a positive gain, while $J^{\dagger}$ is the pseudo-inverse of the Jacobian matrix. Introducing (6) into (4) yields

$$\dot{\mathbf{p}}_i = -J\lambda J^{\dagger}\mathbf{e} = -\lambda\mathbf{e}, \tag{7}$$

considering $\dot{\mathbf{p}}_i \approx \dot{\mathbf{e}} = -\lambda\mathbf{e}$ results in the convergence of $\mathbf{e} \rightarrow 0$.

Considering Assumption 1, then $\omega_x \approx \omega_y \approx 0$, therefore, the desired velocities for the UAV are computed as

$$\begin{bmatrix} v_{d_x} \\ v_{d_y} \\ v_{d_z} \\ \omega_{d_z} \end{bmatrix} = -\lambda \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} J^{\dagger}\mathbf{e}. \tag{8}$$

## 4 SEMI-MARKOV DECISION PROCESS FOR AUTONOMOUS LANDING

Before introducing semi-Markov Decision Processes (SMDP), we will present some necessary mathematical concepts for their understanding, such as the concept of Markov chains.

### *4.1 Markov chains*

A Markov chain is a mathematical model that describes a sequence of random variables where the probability of transitioning to the next state depends only on the current state and not on the previous states (the Markov property). Formally, let $\mathbb{N} := \{0, 1, 2, ..\}$ be the set of natural numbers. The sequence $\{\mathcal{X}_t\}_{t\in\mathbb{N}}$ which takes values in state space $S$ is a Markov chain if it satisfies the following equation:

$$\begin{aligned} &P(\mathcal{X}_{t+1} = j \mid \mathcal{X}_t = i, \mathcal{X}_{t-1} = \cdot, \ldots, \mathcal{X}_0 = \cdot) \\ &= P(\mathcal{X}_{t+1} = j \mid \mathcal{X}_t = i), \qquad \forall\ i, j \in S. \end{aligned}$$

Markov chains are essential for modeling dynamic systems where the next state depends only on the current state. However, as discussed in the introduction, the primary limitation when utilizing Markov chains for system modeling lies in the sojourn time in a state. The sojourn time is a random variable that must be modeled by a geometric distribution (in the discrete case) or by an exponential distribution (in continuous time). To generalize this hypothesis in this article we propose a SMDP to achieve the landing aircraft decision. Before introducing the decisions equations we shall introduce semi-Markov processes.

### *4.2 semi-Markov processes*

Let us consider a complete probability space $(\Omega, \mathcal{F}, \mathbb{P})$ on which all processes and random variables are defined. Let $\{\mathcal{X}_t\}_{t\in\mathbb{N}}$ be a stochastic process which takes values in state space $S$. Let us also define by $\{\tau_n\}_{n\in\mathbb{N}}$ the consecutive time instants when $\{\mathcal{X}_t\}_{t\in\mathbb{N}}$ shifts its state. By definition $\tau_0 := 0$ and

$$\tau_{n+1} := \inf\{t > \tau_n : \mathcal{X}_t \neq \mathcal{X}_{\tau_n}\},\ \ n \geq 0.$$

By convention $\inf \varnothing = \infty$. Observe that sequence $\{\tau_n\}_{n\in\mathbb{N}}$ represents the renewal or jump points of process $\{\mathcal{X}_t\}_{t\in\mathbb{N}}$. We shall denote by $\{B_t\}_{t\in\mathbb{N}}$ the backward process of process $\{\mathcal{X}_t\}_{t\in\mathbb{N}}$. Process $\{B_t\}_{t\in\mathbb{N}}$ is mathematically defined as follows

$$B_t := t - \max\{\tau_n : \tau_n \leq t\} \ \text{ and } B_t := t, \ \text{ if } \tau_1 > t.$$

Speaking generally, the backward process models the time spent in the current state since the last state change.
We define $N(t) := \max\{n \in \mathbb{N} : \tau_n \leq t\}$ as the counting process of the number of jumps in the interval time $[1, t]$ of process $\{\mathcal{X}_t\}_{t\in\mathbb{N}}$. Let us denote by $\{\mathcal{Y}_n\}_{n\in\mathbb{N}}$ a process which records $\{\mathcal{X}_t\}_{t\in\mathbb{N}}$ at jump points $\{\tau_n\}_{n\in\mathbb{N}}$, i.e., $\mathcal{Y}_n = \mathcal{X}_{\tau_n}$ or equivalent $\mathcal{X}_t = \mathcal{Y}_{N(t)}$. We represent by $\{T_n\}_{n\in\mathbb{N}}$ the process that models the time interval between two consecutive jumps. By convention $T_0 := \tau_0$ and $T_{n+1} := \tau_{n+1} - \tau_n$, $n \in \mathbb{N}$. If the following condition is satisfied

$$\begin{aligned} \mathbb{P}(\mathcal{Y}_{n+1} = j,\ T_{n+1} = k \mid \mathcal{Y}_0 = \cdot, \ldots, \mathcal{Y}_n = i;& \\ \tau_0 = \cdot, \ldots, \tau_n = \cdot)& \\ = \mathbb{P}(\mathcal{Y}_{n+1} = j,\ T_{n+1} = k \mid \mathcal{Y}_n = i)& \quad (9) \end{aligned}$$

then process $\{\mathcal{X}_t\}_{t\in\mathbb{N}}$ is a semi-Markov chain (SMC) and process $\{\mathcal{Y}_n\}_{n\in\mathbb{N}}$ is know as the embedded Markov chain of process $\{\mathcal{X}_t\}_{t\in\mathbb{N}}$. From now on to make the notation less cumbersome, instead of writting $\{\mathcal{X}_t\}_{t\in\mathbb{N}}$ we will write $\{\mathcal{X}_t\}$ and apply this simplification to all processes.
If the right-hand side of Equation (9) does not depend on $n$ then $\{\mathcal{X}_t\}$ and $\{\mathcal{Y}_t\}$ are time homogeneous. In this article we shall consider a homogeneous discrete-time semi-Markov process $\{\mathcal{X}_t\}$. A discrete-time semi-Markov process is also referred to as a semi-Markov chain (SMC). A SMC is fully specified by its state space, in this case it will be denoted by $S$ and, its semi-Markov kernel $\mathbf{K}(t) = \{K_{ij}(t);\ \ i, j \in S, t \in \mathbb{N}\}$ where

$$K_{ij}(t) := \mathbb{P}(\mathcal{Y}_{n+1} = j, T_{n+1} = k \mid \mathcal{Y}_n = i), \quad k \geq 0,\ t \in \mathbb{N}. \tag{10}$$

The cumulative semi-Markov kernel of SMC $\{\mathcal{X}_t\}$ is defined by

$$\begin{aligned} Q_{ij}(t) \ &:= \ \mathbb{P}(\mathcal{Y}_{n+1} = j, T_{n+1} \leq t \mid \mathcal{Y}_n = i) \\ &= \ \sum_{k=0}^{t} K_{ij}(k),\ \ i, j \in S, t \in \mathbb{N}. \end{aligned}$$

The cumulative distribution function of the sojourn time in state $i \in S$ is defined by

$$H_i(t) := \sum_{k=0}^{t} \sum_{j\in S} K_{ij}(k)$$

The conditional cumulative distribution of the sojourn time is

$$\begin{aligned} F_{ij}(t) := \mathbb{P}(T_{n+1} \leq t \mid J_n = i, J_{n+1} = j)& \\ = \begin{cases} \frac{Q_{ij}(t)}{p_{ij}}, & \text{if } p_{ij} \neq 0, \\ 1, & \text{if } p_{ij} = 0. \end{cases}& \quad (11) \end{aligned}$$

The primary distinction between Markov and semi-Markov discrete-time processes lies in the distribution function $F_{ij}(t)$. For a Markov chain, this function follows a geometric distribution with a success parameter of $1 - P(i, i)$, where $P$ represents the Markov transition probability matrix. Conversely, in a semi-Markov chain, the distribution function $F_{ij}(t)$ may be represented by any discrete-time distribution.
Throughout this article, note that we employ the index $t \in \mathbb{N}$ to denote calendar time, while the index $n \in \mathbb{N}$ is used for the renewal points of SMC $\{\mathcal{X}_t\}$.

In this subsection, we provided a summary of the fundamental background on a SMC. In the following subsection, we will present the Markov Decision Process (MDP). After that we shall generalize the MDP into SMDP.

### *4.3 Markov Decision Process (MDP)*

A MDP is a mathematical framework used for modeling decision-making situations where outcomes are partly random and partly under the control of a decision maker. It consists of the following components:

- States $S$: A set of possible situations or configurations the system can be in.
- Actions $A$: A set of possible actions the decision maker can take from a state.
- Transition probabilities between states: The probability of transitioning from one state to another.
- Rewards $R$: A numerical value received after taking an action in a state, representing the immediate benefit or cost of that action.
- Discount factor $\gamma$: A constant number ($0 \leq \gamma \leq 1$). If $\gamma \approx 1$ the system prioritiza future rewards by contrast if $\gamma \approx 0$ the system prioritize immediate rewards.
- Policy $\pi$: A strategy or a plan that defines the action to be taken in each state.

The goal of a MDP is to find an optimal policy that maximizes the expected value of long-term rewards. To achieve this goal, Bellman equation is used.

The Bellman equation is a fundamental recursive equation in dynamic programming and reinforcement learning, see for e.g., [13]. It helps to determine the optimal policy in decision-making problems. The Bellman equation is typically written as

$$V_\pi(s) = \max_a \left[ \sum_{s'} P(s'|s,a) \left( R(s,a) + \gamma V_\pi(s') \right) \right], \quad (12)$$

where

- $V_\pi(s)$ is the value function, representing the expected cumulative reward from state $s$ applying policy $\pi$.
- $R(s,a)$ is the immediate reward received after taking action $a$ in state $s$.
- $\gamma$ is the discount factor (between 0 and 1), which determines the importance of future rewards.
- $P(s'|s,a)$ is the transition probability of reaching state $s'$ from $s$ given action $a$.
- $\pi$ is a plan or strategy that outlines the actions to be executed in every given state.

The idea of MDP is to chose a policy $\pi^*$ such that

$$\pi^*(s) = \arg\max_a \left[ \sum_{s'} P(s'|s,a) \left( R(s,a) + \gamma V_{\pi^*}(s') \right) \right]. \quad (13)$$

In others words, Bellman equation maximizes the expected value $V_\pi(s) = \mathbb{E}[G_n | \mathcal{X}_n = i]$, where $G_n := \sum_{k=0}^{\infty} \gamma^k R_{n+k+1}$.

Thus far, we have outlined the equations modeling MDPs, but these models have certain limitations. In particular, the sojourn time within a state is a random variable modeled by a geometric distribution (in the discrete case). Therefore, the next subsection will be introduced SMDPs, where the sojourn time can be represented by any distribution function.

### *4.4 semi-Markov Decision Process (SMDP)*

MDP have some disadvantages respect of the distribution function of the sojourn time in a state, for this reason in this article we propouse a SMDP to successfully achieve the landing of a UAV on a moving platform.
It is well known that if process $\{\mathcal{X}_t\}$ is an SMC then the bivariate process $\{\mathcal{X}_t, B_t\}$ is a Markov chain, see, e.g., [17], where its transition probability matrix $\mathcal{P}$ is a function of the semi-Markov kernell as follows

$$\mathcal{P}(i,b_1)(j,b_2) = \begin{cases} \frac{q_{i,j}(b_1+1)}{\overline{H}_i(b_1)}, & \text{if } i \neq j \text{ and } b_2 = 0; \\ \frac{\overline{H}_i(b_1+1)}{\overline{H}_i(b_1)}, & \text{if } i = j \text{ and } b_2 - b_1 = 1. \end{cases} \quad (14)$$

where $\overline{H}_i(b) := \mathbb{P}(T_1 > b \mid \mathcal{Y}_0 = i) = 1 - H_i(b)$.

We use bivariate process $\{\mathcal{X}_t, B_t\}$ to select the best action according with the state and the backward of the system, in others words, we search a policy $\pi^{*'}$ such that

$$\pi^{*'}(s) = \arg\max_a \left[ \sum_{s'} P(s'|s,b,a) \left( R(s,b,a) + \gamma V_{\pi^{*'}}(s') \right) \right]. \quad (15)$$

The main difference between Equiations (13) and (15) is that the transition probabilities and rewards now depend on the time the system has spent in the present state which is a random variable, modeled by any discrete time distribution function.

### *4.5 Implementation*

A Semi-Markov Decision Process (SMDP) is employed to enable UAV landing on a moving platform. The system defines five discrete states and five possible actions, summarized in Table 1.

Table 1: SMDP States and Actions

| State | Description | Action | Description |
|---|---|---|---|
| $S1$ | Platform not in sight | $A1$ | Move in circles |
| $S2$ | Platform in sight | $A2$ | Center code |
| $S3$ | Platform below | $A3$ | Move forward |
| $S4$ | Landed (fail) | $A4$ | Move backward |
| $S5$ | Landed (success) | $A5$ | Land |

Available actions are $\{A1, A3, A4, A5\}$ in $S1$, and $\{A1, A2, A3, A4, A5\}$ in $S2$ and $S3$. States $S4$ and $S5$ are absorbing; any action leads back to the same state with probability one.

Rewards guide the UAV's decisions by encouraging progress and penalizing undesired behavior. The reward function is empirically set as:

- Progress to next state: $+3$
- Remain in current state: $+1$
- Return to previous state: $-3$

In a SMC the semi-Markov kernel compuetes transition probabilities between states, see Equation (10). The empirical estimator for the semi-Markov kernel used in this article is given by following equation

$$\hat{K}_{ij}(t, M) := \frac{N_{ij}(t, M)}{N_i(M)} \quad \text{for } i, j \in S; t \in \mathbb{N}, \tag{16}$$

where $M \in \mathbb{N}^*$ is a sensor time and $N_{ij}(t, M)$ is the number of transitions on the embedded Markov chain $\{\mathcal{Y}_t\}$ from $i$ to $j$ up to time $M$, with sojourn time $t$ such that $1 \leq t \leq M$. $N_{ij}(t, M)$ is formally defined by

$$\begin{aligned} N_{ij}(t, M) &:= \sum_{n=1}^{N(M)} \mathbf{1}_{\{\mathcal{Y}_{n-1}=i, \mathcal{Y}_n=j, T_n=t\}} \\ &= \sum_{n=1}^{M} \mathbf{1}_{\{\mathcal{Y}_{n-1}=i, \mathcal{Y}_n=j, T_n=t, \tau_n \leq M\}}. \end{aligned}$$

Therefore

$$N_i(M) := \sum_{t=0}^{\infty} \sum_{j \in S} N_{ij}(t, M)$$

Notice that $\mathbf{1}_{\{x=A\}}$ is the indicator function of event $A$, that means

$$\mathbf{1}_{\{x=A\}} = \begin{cases} 1 & if \;\; x = A; \\ 0 & \text{otherwise.} \end{cases}$$

The data for computing Equation(16) where taken from several flights.

### *4.6 Algorithm Implementation*

The following steps outline the implementation of the mathematical framework described in Section 4:

1. For every $s \in S$ initialize $V_\pi(s)$, with $V_\pi(s) = 0$ for non-absorbing states.
2. Iteratively update $V_\pi(s)$ using the Bellman equation.
3. Compute the optimal policy $\pi^{*'}(s)$ based on $V_\pi(s)$.

## 5 EXPERIMENTAL VALIDATION

Experimental validation is a key component of this study. The UAV states, actions, and rewards are defined based on manual landing trials. A DJI Tello UAV is selected for its compact size, efficiency, and ease of programming.

In the first experimental stage, landings on a platform moving at constant linear velocity in one dimension are performed. The UAV's position is adjusted to match the platform's motion, while manual control is applied to simulate approach, alignment, descent, and touchdown phases. Each maneuver is associated with specific actions and rewards to reinforce desired behaviors.

The collected data is used to refine the SMDP model, enabling more robust decision-making for autonomous landing.

After the first stage in experimentation the following computations were obtained based on the provided transition data.

### *5.1 Transition Probability and Reward Matrices*

Table 2 shows a sample of the derived transition probability ($P$) and rewards ($R$) of the embedded Markov chain $\{\mathcal{Y}_t\}$ of SMC $\{\mathcal{X}_t\}$ of the UAV.

Table 2: Transition Probability and Reward Matrices

| State | Action | Next State ($s'$) | $P(s'\|s, a)$ | $R(s, a)$ |
|---|---|---|---|---|
| 1 | 1 | 2 | 0.0 | 1.0 |
| 1 | 1 | 3 | 0.0 | 1.0 |
| 2 | 2 | 3 | 1.0 | 3.0 |
| 3 | 3 | 4 | 1.0 | -3.0 |

### *5.2 Value Function and Optimal Policy*

The value function $V_\pi(s)$ after convergence and the optimal policy $\pi^*(s)$ are shown in Table 3. This table helps the UAV understand the best decision based on its actual state.

Table 3: Value Function $V_\pi(s)$ and Optimal Policy $\pi^*(s)$

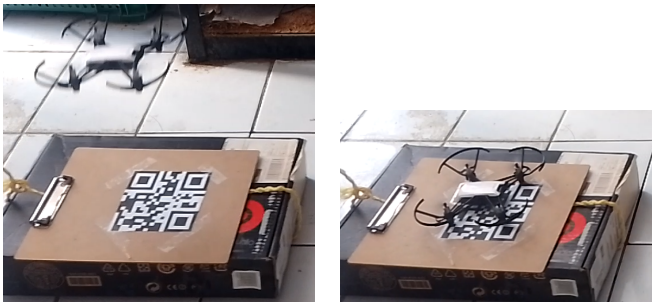
| State ($s$) | Value ($V(s)$) | Optimal Action ($a$) |
|---|---|---|
| 1 | 12.76 | 1 |
| 2 | 11.76 | 5 |
| 3 | 9.99 | 3 |
| 4 | 0.0 | – |
| 5 | 0.0 | – |

### 5.3 Results

The experimental results demonstrate a progressive improvement in the UAV's landing performance as the SMDP model undergoes training and parameter adjustments. Initial landing attempts showed a high failure rate, highlighting the need for fine-tuning both the decision-making framework and control parameters. However, as the system iteratively learned from previous experiences, the success rate significantly increased, reflecting the model's ability to adapt and optimize its performance over time.

By the end of the experimental phase, the UAV consistently achieved successful landings with greater precision and stability. To further illustrate the system's evolution and final performance, a demonstration video is provided, showcasing representative landing sequences throughout the training process:
`https://youtu.be/FTALlxdZ64E`.

Some screenshots have been taken from the video and depicted in Figures 3a and 3b



(a) Drone approaching (b) Drone landing

Figure 3: Drone landing procedure

Following figures represent the results of a series of flights. The first graph represents a comparative analysis of the percentage of successful drone landings over five-flight intervals using three different control strategies: manual landing (Non-Markov), Markov, and Semi-Markov processes.

The experiment involved grouping 25 drone flights into five segments of five flights each and calculating the success rate for each method in every interval. The Non-Markov approach, which lacks probabilistic decision-making, showed lower and inconsistent success rates. The Markov approach, which uses memoryless transitions between states, performed better but had fluctuations.

The Semi-Markov approach, which considers both state transitions and time delays, achieved the highest and most stable success rates as depicted in Figure 4.

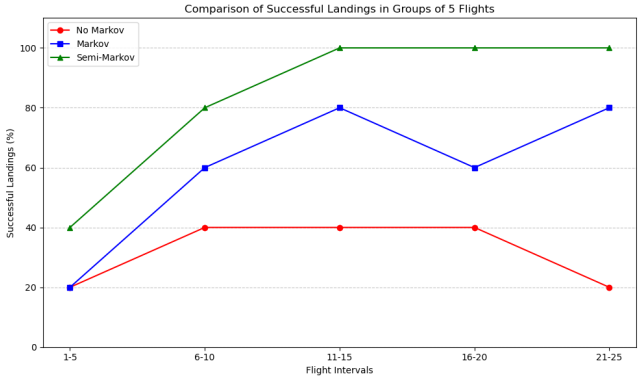



Figure 4: Comparison of successful landings

The effectiveness of three different landing strategies (Non-Markov, Markov, and Semi-Markov) is evaluated by measuring the closest distance to the platform across five flight trials, see Figure 5. Each strategy was applied to a drone during landing attempts, and for each flight, the shortest distance achieved from the target at the final stage of the flight was recorded. The improvement in landing accuracy over successive flights is represented in the graph, with the Y-axis indicating the closest distance to the platform (0–30 cm, where lower is better) and the X-axis showing the flight number. It is observed that the most precise landings are consistently achieved with the Semi-Markov strategy, followed by the Markov strategy, while the Non-Markov approach shows the least improvement over time. These results suggest that the drone's landing accuracy is enhanced over repeated attempts when probabilistic decision-making models, particularly Semi-Markov processes, are incorporated.

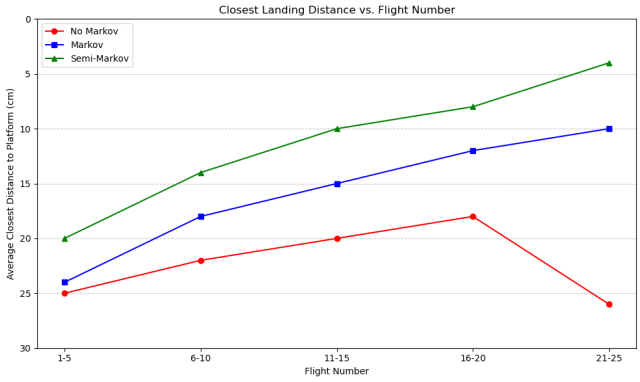



Figure 5: Comparison of distance from target

## 6 Conclusion

This paper demonstrates the effectiveness of Markov Decision Processes (MDP) and Semi-Markov Decision Processes (SMDP) for UAV autonomous landing. These models provide robust tools for real-time decision-making under

uncertainty, as validated by experimental results showing improved performance over non-probabilistic approaches.

The system exhibited notable gains in robustness and decision accuracy. Initially, high failure rates highlighted the challenges of dynamic environments, but iterative learning led to progressively higher landing success, reflecting the adaptability of the SMDP framework.

A key advantage of SMDPs was their capacity to model temporal dynamics, outperforming standard MDPs by accounting for state transitions and time delays. This resulted in greater landing precision and stability.

Future work will focus on mitigating the system's tendency to remain in certain states too long, improving responsiveness via enhanced temporal modeling and adaptive learning. Testing in more complex, unpredictable scenarios will further assess scalability and robustness.

Probabilistic models (particularly SMDPs) prove valuable for improving UAV autonomy, offering a path toward more reliable and efficient real-world deployment.



## ACKNOWLEDGEMENTS

The authors would like to express their gratitude to the Computer Science Department at Tecnológico de Monterrey, Guadalajara Campus, for providing the resources and institutional support that made this research possible.

## REFERENCES

[1] Alvika Gautam, PB Sujit, and Srikanth Saripalli. A survey of autonomous landing techniques for uavs. In *2014 international conference on unmanned aircraft systems (ICUAS)*, pages 1210–1218. IEEE, 2014.

[2] Xiaoshan Bai, Youqiang Ye, Bo Zhang, and Shuzhi Sam Ge. Efficient package delivery task assignment for truck and high capacity drone. *IEEE Transactions on Intelligent Transportation Systems*, 2023.

[3] Mayank Mittal, Rohit Mohan, Wolfram Burgard, and Abhinav Valada. Vision-based autonomous uav navigation and landing for urban search and rescue. In *The International Symposium of Robotics Research*, pages 575–592. Springer, 2019.

[4] Jonathan Sprinkle, J Mikael Eklund, and S Shankar Sastry. Deciding to land a uav safely in real time. In *Proceedings of the 2005, American Control Conference, 2005.*, pages 3506–3511. IEEE, 2005.

[5] François Chaumette. Visual servoing. In *Computer Vision: A Reference Guide*, pages 1367–1374. Springer, 2021.

[6] Francois Chaumette. Potential problems of stability and convergence in image-based and position-based visual servoing. In *The confluence of vision and control*, pages 66–78. Springer, 2007.

[7] Efstratios Kakaletsis, Charalampos Symeonidis, Maria Tzelepi, Ioannis Mademlis, Anastasios Tefas, Nikos Nikolaidis, and Ioannis Pitas. Computer vision for autonomous uav flight safety: An overview and a vision-based safe landing pipeline example. *Acm Computing Surveys (Csur)*, 54(9):1–37, 2021.

[8] Daewon Lee, Tyler Ryan, and H Jin Kim. Autonomous landing of a vtol uav on a moving platform using image-based visual servoing. In *2012 IEEE international conference on robotics and automation*, pages 971–976. IEEE, 2012.

[9] Florent Le Bras, Tarek Hamel, Robert Mahony, Christian Barat, and Julien Thadasack. Approach maneuvers for autonomous landing using visual servo control. *IEEE transactions on aerospace and electronic systems*, 50(2):1051–1065, 2014.

[10] Pedro Serra, Rita Cunha, Tarek Hamel, David Cabecinhas, and Carlos Silvestre. Landing of a quadrotor on a moving target using dynamic image-based visual servo control. *IEEE Transactions on Robotics*, 32(6):1524–1535, 2016.

[11] Alessandra Elisa Sindi Morando, M Ferreira Santos, P Castillo, and A Correa-Victorino. Vision-based algorithm for autonomous aerial landing. In *2024 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 652–657. IEEE, 2024.

[12] Martin L Puterman. Markov decision processes. *Handbooks in operations research and management science*, 2:331–434, 1990.

[13] Richard S Sutton. Reinforcement learning: An introduction. *A Bradford Book*, 2018.

[14] Riccardo Polvara, Massimiliano Patacchiola, Sanjay Sharma, Jian Wan, Andrew Manning, Robert Sutton, and Angelo Cangelosi. Toward end-to-end control for uav autonomous landing via deep reinforcement learning. In *2018 International conference on unmanned aircraft systems (ICUAS)*, pages 115–123. IEEE, 2018.

[15] Jacques Janssen and Nikolaos Limnios. *Semi-Markov models and applications*. Springer Science & Business Media, 2013.

[16] Odile Bourquardez, Robert Mahony, Nicolas Guenard, François Chaumette, Tarek Hamel, and Laurent Eck. Image-based visual servo control of the translation kinematics of a quadrotor aerial vehicle. *IEEE Transactions on Robotics*, 25(3):743–749, 2009.

[17] Vlad Stefan Barbu and Nikolaos Limnios. *Semi-Markov chains and hidden semi-Markov models toward applications: their use in reliability and DNA analysis*, volume 191. Springer Science & Business Media, 2009.