# Autonomous UAV Navigation for Active Perception of Targets in Uncertain and Cluttered Environments

**Juan Sandino**
**School of Electrical Engineering and Robotics**
**Queensland University of Technology**
**2 George St., Brisbane City, QLD 4000, Australia**
**j.sandino@qut.edu.au**

**Fernando Vanegas**
**School of Electrical Engineering and Robotics**
**Queensland University of Technology**
**2 George St., Brisbane City, QLD 4000, Australia**
**f.vanegasalvarez@qut.edu.au**

**Felipe Gonzalez**
**School of Electrical Engineering and Robotics**
**Queensland University of Technology**
**2 George St., Brisbane City, QLD 4000, Australia**
**felipe.gonzalez@qut.edu.au**

**Frederic Maire**
**School of Electrical Engineering and Robotics**
**Queensland University of Technology**
**2 George St., Brisbane City, QLD 4000, Australia**
**f.maire@qut.edu.au**

***Abstract*—The use of Small Unmanned Aerial Vehicles (sUAVs) has grown exponentially owing to an increasing number of autonomous capabilities. Automated functions include the return to home at critical energy levels, collision avoidance, take-off and landing, and target tracking. However, sUAVs applications in real-world and time-critical scenarios, such as Search and Rescue (SAR) is still limited. In SAR applications, the overarching aim of autonomous sUAV navigation is the quick localisation, identification and quantification of victims to prioritise emergency response in affected zones. Traditionally, sUAV pilots are exposed to prolonged use of visual systems to interact with the environment, which causes fatigue and sensory overloads. Nevertheless, the search for victims onboard a sUAV is challenging because of noise in the data, low image resolution, illumination conditions, and partial (or full) occlusion between the victims and surrounding structures. This paper presents an autonomous Sequential Decision Process (SDP) for sUAV navigation that incorporates target detection uncertainty from vision-based cameras. The SDP is modelled as a Partially Observable Markov Decision Process (POMDP) and solved online using the Adaptive Belief Tree (ABT) algorithm. In particular, a detailed model of target detection uncertainty from deep learning-based models is shown. The presented formulation is tested under Software in the Loop (SITL) through Gazebo, Robot Operating System (ROS), and PX4 firmware. A Hardware in the Loop (HITL) implementation is also presented using an Intel Myriad Vision Processing Unit (VPU) device and ROS. Tests are conducted in a simulated SAR GPS-denied scenario, aimed to find a person at different levels of location and pose uncertainty.***

## TABLE OF CONTENTS



## 1. INTRODUCTION

Unmanned Aerial Vehicles (UAVs) are currently used in significant civilian applications such as remote sensing, disaster monitoring, surveillance, and Search and Rescue (SAR) [1]. Part of the success of this technology is caused by cheaper hardware and advances in sensor systems, computer vision and image processing, and autonomous navigation. Advances in autonomous navigation include automated home return at critical battery levels, reactive collision avoidance, autonomous take-off and landing, and active target tracking. Nonetheless, the deployment of these systems at a broader scale is still restricted by operational limitations in hardware and software [2]. Current limitations on Small UAVs (sUAVs are drones that weigh less than 13 $kg$ [3]) include: (a) onboard computing power, (b) payload weight, (c) energy storage, (d) sensor resolution and image quality and (e) cognition capabilities in unstructured environments [4].

While recent research suggests that some hardware constraints are likely to be resolved soon [5], the development of autonomous decision-making processes on sUAVs is a far more challenging problem [4]. Decision-making capabilities on sUAVs are still limited when dealing with mid-flight events, path planning and obstacle avoidance, or target finding under uncertainty and partial observability [6]. Engineers and roboticists usually reduce uncertainty by adjusting robot working environments to become as structured as possible [7]. However, real-world applications are unstructured, full of uncertainties.

Natural disasters are events that unfortunately still claim human lives around the globe [8, 9]. Emergency situations also occur by unfortunate situations such as people getting into distress in rivers and shores and lost people in rural areas. Disaster management and rescue workers are critical to react in emergency situations and diminish fatalities. In SAR applications, the overarching aim of autonomous sUAV navigation is the quick localisation, identification and quantification of victims to prioritise emergency response in affected zones [10]. Various challenges need to be addressed to enable a sUAV to navigate autonomously in SAR scenarios.

In emergency situations, the available information for the evaluation of access areas, affected structures (if any), and the identification of victims (if any) is usually unknown or limited [11]. Second, the vision tasks performed by a sUAV are challenging because of noise in the data, low image resolution, illumination conditions, and partial (or full) occlusion of the victims. Additionally, the application of sUAVs under

cluttered and GPS-denied environments require the use of Simultaneous Localisation and Mapping (SLAM) algorithms that rely on advanced sensor systems (*e.g.* LiDAR), which are often expensive, complex to operate and sometimes, computationally intensive. Moreover, optimal control of sUAVs require the use of workstations owing to their resource-constrained onboard hardware. Normally, sUAV operators decide on the next sequence of navigation commands to interact with the environment using visual telemetry [12]. However, the prolonged use of these interactive systems is claimed to produce fatigue and sensory overloads. Last, the sUAV must avoid collisions within the surveyed areas, it should generate a path-planning strategy for exploration based on the environment, natural disturbances (*e.g.* wind, atmospheric pressure, temperature) and the drone kinematic constraints.

Despite recent advances in autonomous sUAVs for SAR applications [13–15], there are still unresolved issues. Namely, lack of validation of these systems in real-world environments, uncertainty in the identification, location and counting of objects of interest (*i.e.* victims) under complex image representations, and optimal interaction between the drones and the victims. The above-mentioned limitations motivate the exploration of sUAV systems with increased cognition and autonomy level in stochastic environments under target detection uncertainty from vision-based sensors. This paper presents a framework for autonomous sUAV navigation that models target location and identification uncertainties from vision-based cameras and object detection models. The problem is formulated as a Partially Observable Markov Decision Process (POMDP) and solved online using the Adaptive Belief Tree (ABT) algorithm. In particular, a detailed model of the uncertainty derived from a deep learning object detector algorithm is demonstrated. The presented system handles autonomous sUAV navigation under uncertainty as a multi-objective problem comprising path planning, obstacle avoidance, motion control, and target detection tasks. The proposed approach addresses the following challenges:

1. Autonomous sUAV decision-making under environment uncertainty in SAR applications.
2. Uncertainty on the identification and location of victims in cluttered and GPS-denied environments.
3. Modelling of target detection uncertainty from vision-based sensors.
4. Execution of computationally expensive decision-making and object detection algorithms in resource-constrained hardware.

The proposed system is validated with experiments in simulation. In these experiments, the sUAV is tasked with the mission of finding a lost person in a cluttered environment.

The rest of paper is structured as follows: Section 2 discusses previous works on autonomous sUAV decision-making under uncertainty and target detection uncertainty. Section 3 reviews briefly uncertainty representation, POMDPs and online ABT solvers for autonomous sUAV navigation. Section 4 details the problem formulation and modelling of target detection uncertainty. Section 5 describes the proposed system architecture, software and hardware tools, and experimentation setup. The evaluation of the suggested formulation under a simulated cluttered and GPS-denied scenario is shown in Section 6. Finally, Section 7 discusses future work.

## 2. Related Work

Theory on decision-making is extensive and relates not only to autonomous sUAV navigation but also to other fields such as multi-objective decision-making, game theory, navigation strategies, Bayesian principles, Markov Decision Processes (MDPs) and Partially Observable Markov Decision Processes (POMDPs) [16]. Since time-critical applications (*e.g.* SAR) feature uncertainty and partial observability in the system states and targets, MDPs and specially POMDPs have proven to be useful while making navigation decisions under these conditions [10, 17–20]. Literature has likewise shown how the modelling of POMDPs in highly uncertain environments and partial observability is a suitable approach for sUAV navigation problems [16]. Vanegas and Gonzalez [21], for example, implemented an autonomous sUAV navigation algorithm for GPS-denied and cluttered environments. The authors compared two of the fastest POMDP online solvers, namely Partially Observable Monte Carlo Planning (POMCP) [22] and Adaptive Belief Tree (ABT) [23]. The proposed framework detailed the possibility of using the ABT solver for the drone to make decisions in seconds. Nonetheless, the authors narrowed their tests for indoor environments and detected trivial targets, particularly 2D markers. Additionally, the navigation commands were transmitted using the sUAV communication module to a workstation. As discussed by Valavanis and Vachtsevanos [24] and Carrio et al. [4], the dependency of communication modules for these tasks is undesirable because the drone's behaviour under complex environments might become seriously compromised if those modules fail.

Another significant research by Ragi and Chong [18, 25] presents POMDP-based solvers for dynamic path-planning applied to multiple target tracking. The POMDP formulation became more significant towards fully autonomous sUAVs by including path planning, collision avoidance, external wind disturbance effects, and tracking evasive treats. Similarly, Bravo et al. [10] assessed POMDP frameworks in humanitarian relief applications through simulation. The authors concluded a higher need to validate existing methods in real disaster situations. Other related POMDP-based solvers have also shown advances compared to standard models such as Anytime Meta PLannEr (AMPLE) [26], POMDP-lite [27], decentralised POMDP [28] and Mixed Observability Markov Decision Process (MOMDP) [29].

Research studies on onboard autonomous sUAV decision-making for real-time applications using POMDPs are limited. One of the most notable studies is the work conducted by Chanel et al. [30], who could show a multi-target car recognition application using a customised optimisation framework. Their system was able to run the POMDP solver onboard the sUAV and optimised during execution. The authors, nevertheless, did not provide relevant experimentation details such as the sUAV model, hardware specifications and vision-based algorithms. Furthermore, they also demonstrated the framework in an open rural field, where levels of target uncertainty could become considerably low once a car is perceived under the camera's FOV and detected by vision-based detectors. Compared to emergency environments, image representations of victims are more challenging due to partial or full occlusion, pose, and dynamics.

## 3. BACKGROUND

*Types of Uncertainty*

sUAV perception is limited by noisy sUAV onboard sensors, poor image representations of the targets caused by partial and entire occlusion from other objects, and the dynamics of the target itself [7]. Partial observability from vision-based cameras introduces target uncertainty, which can be classified as follows:

1. *Location:* target location uncertainty is caused by the limited extent of vision-based sensors to capture images (also known as Field of View (FOV)). These limitations are present in mapping, surveillance, and any other target finding applications under cluttered and challenging environments. The location of victims in SAR operations is, for example, challenging when vision-based sensors cannot cover specific FOV configurations (*e.g.* sUAV viewpoint in non-nadir or side-by-side settings) or when the victims are partially occluded by an urban structure, the effects of the disaster situation (*e.g.* fire smokes, earthquakes, floods) or natural events (*e.g.* fog, cloudy and wet conditions).
2. *Identification:* this refers to limitations on recognising a specific identification property of the object of interest. Identification uncertainty on victims includes age, gender and health conditions.
3. *Quantity:* target counting uncertainty applies where multiple victims are partially occluded among themselves, between themselves and environment objects, or environmental factors that affect the image quality of vision-based sensors. Counting is important in humanitarian relief operations when rescuers are required to prioritise emergency response by estimating the number of victims. The collection of these statistics will determine, therefore, the areas that require immediate intervention.
4. *Dimensions:* this type of uncertainty occurs when the surveying application requires the collection of morphological properties of the victims. Examples include a person's height and volume. In SAR operations, the awareness of a victim's height or volume could infer complementary rescue conditions and define resources and processes needed to assist them.

Another source of uncertainty comes from the outputs of computer vision algorithms for object detection. Developed algorithms use varied strategies for object detection, ranging from classical image processing manipulations (*e.g.* filtering, thresholding, edge and contour detection, and morphological operations), feature extraction (*e.g.* SIFT, SURF), adaptive filtering (*e.g.* Kalman filters), visual odometry (*e.g.* SLAM) and deep learning (*e.g.* Region-based and Single Shot Detector CNNs) [31]. SDPs for autonomous sUAV decision-making should consider uncertainty from vision-based target detectors to adjust their path planning when detections lack accuracy.

Uncertainty in sUAV navigation is caused by the kinematics of the drone and disturbances from the environment. Since a sUAV takes off, there is always a present drift in the initial position and heading. Errors in a sUAV local position estimator likewise exist from propagated errors from sensor readings and motion controllers. Last, the dynamics in the environment might affect the behaviour of sUAVs, such as changes in wind direction and currents, temperature and atmospheric pressure.

## 4. PROBLEM FORMULATION

*Partially Observable Markov Decision Process*

The process of adding cognition capabilities to a sUAV in uncertain environments can be catalogued as a decision-making problem. An autonomous sUAV system should decide the optimal sequence of actions that maximises the probabilities of accomplishing the flight mission. The flight mission consists in searching for a victim and stops once the first victim is found. This selection of sequence of actions are evaluated not only by achieving the primary search goal, but also by following some interaction rules with the environment, namely collision avoidance, the exploration within a pre-defined Region of Interest (ROI) and short flight times. Considering that real-world environments are dynamic, the sUAV should also evaluate changes in the environment and update its path planning strategy. Therefore, there must be a constant interaction between the sUAV and the environment, where each action taken by the drone at a time $t$ should be monitored using collected data from the environment in the form of observations. This data analysis aims to infer the current conditions (or states) of the system and how close the sUAV is to achieve the primary mission goal. Based on that analysis, the sUAV should decide on the next taken action at a time $t+1$ that increases the chances of accomplishing the mission. This optimisation problem aims to define a sequence of actions that maximises those rewards in the long run, which depend on the current state of the environment and individual collected rewards.

MDPs are discrete-time mathematical models that allow the description of Sequential Decision Processes (SDPs) on environments under uncertainty [7]. A problem formulation under MDPs assumes that the system states are *fully* observable. Conversely, POMDPs incorporate uncertainty and *partial* observability from the agent (*i.e.* sUAV) in the system states. POMDPs are defined by the tuple $\langle S, A, T, R, O, \mathcal{Z}, \gamma \rangle$ where $S$ is a finite set of states, $A$ is a finite set of actions, $T$ is a state transition probability matrix $T^a_{ss'} = \mathbb{P}(S_{t+1} = s' \mid S_t = s, A_t = a)$, $R$ is a reward probability matrix $\mathcal{R}^a_s = \mathbb{E}\left[R_{t+1} \mid S_t = s, A_t = a\right]$, $O$ is a finite set of observations, $\mathcal{Z}$ is an observation probability matrix $\mathcal{Z}^a_{s'o} = \mathbb{P}\left[O_{t+1} = o \mid S_{t+1} = s', A_t = a\right]$, and $\gamma$ is a discount factor $\gamma \in [0, 1]$.

In POMDPs, uncertainty in the system states are represented by a probability distribution of the system over all possible states in its state space, called belief states $b$, defined in Equation 1:

$$b(h) = (\mathbb{P}\left[S_t = s^1 | H_t = h\right], \ \cdots, \ \mathbb{P}\left[S_t = s^n | H_t = h\right]) \tag{1}$$

where $H$ is the *history* of actions, observations and rewards that the agent has experienced until time $t$,

$$H_t = a_0, o_1, r_1, \ \cdots, a_{t-1}, o_t, r_t \tag{2}$$

Given a current belief state $b$, the goal of any POMDP solver is to find a sequence of actions that maximises the discounted accumulated reward. This sequence of actions is commonly known as the policy $\pi$. The behaviour of an agent is represented by mapping a policy $\pi : b \rightarrow A$. The POMDP is solved by finding the optimal policy $\pi^*$ that maximises the expected accumulated reward.

$$\pi^* := \arg\max_{\pi} \left( \mathbb{E} \left[ \sum_{\tau=0}^{\infty} \gamma^{t_\tau} R\left(S^{t_\tau}, \pi\left(b^{t_\tau}\right)\right) | b^{t_0}, \pi \right] \right) \quad (3)$$

*Adaptive Belief Tree*

Many online POMDP solvers recompute the optimal policy at each time step from scratch. This concept or re-planning is inefficient for real-time applications because of the loss of computational and time resources by discarding computed policies at previous time steps. While that loss might not be as important on static, structured environments, real-world environments often present gradual or partial changes, which aggravate that loss of resources. The ABT solver, developed by Kurniawati and Yadav [23], proposes the reuse of the previous computed policy and generates policy updates when changes in the POMDP model are detected. Similar to Partially Observable Monte Carlo Planning (POMCP), ABT can approximate optimal policies in continuous state spaces. The ABT solver approximates the solution by maintaining a set of multiple sample episodes. Consequently, the probability distributions of the POMDP are not explicitly defined but as a generative model.

The ABT solver uses an approach of planning and execution in real time. First, an offline optimal policy is calculated based on the POMDP model. Then, the agent executes an action from the offline obtained policy. The agent collects an observation following the chosen action. Afterwards, the belief states are updated based on the collected observation. Subsequently, the ABT solver will update the policy from the updated belief states. Finally, the agent is ready to execute the next action from the updated policy.

*sUAV Navigation Task*

The proposed approach aims for a sUAV to find a lost person in a cluttered indoor environment. The environment, illustrated in Figure 1, contains a restricted flying area, several obstacles and the person to be found at different position configurations (further setup details can be found in Section 5 and Section 6). The sUAV is assumed to incorporate a visual odometry system for pose and motion estimation. Observations from vision-based sensors comprise a downward-looking camera. Functions such as take-off, landing, and return home are delegated to the sUAV autopilot. Therefore, the autonomous sUAV navigation task will begin after the drone reaches an initial waypoint location and will finish after the victim is found. The optimal policy to be learnt is the one that allows the sUAV to accomplish path planning, obstacle avoidance, and finding the victim successfully at different levels of uncertainty. Further details on the system architecture can be found in Section 5.

*State Space*

The state space $S$ is the Cartesian product of $S_r$, the state space of the sUAV $S_v$, the state space of the victim.

$$S = (S_r, S_t) \quad (4)$$

The sUAV states are defined in Equation 5,

$$S_r = (p_r, o_r, f_c, f_b) \quad (5)$$

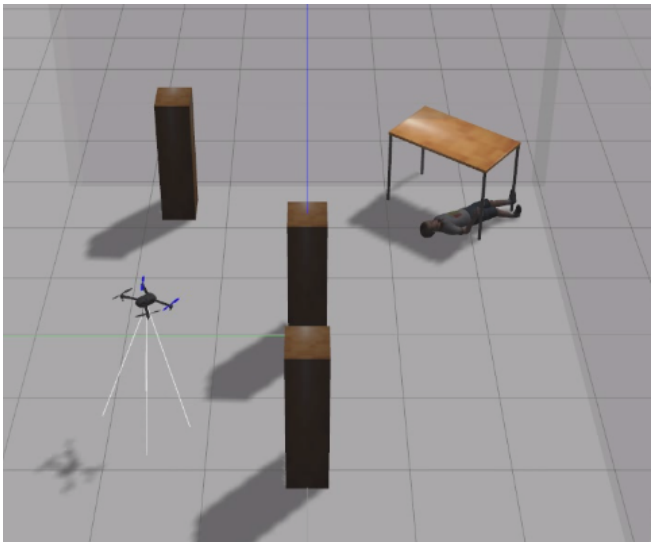


**Figure 1**: **Illustration of the sUAV navigation task to find people in a cluttered environment.**

$$p_r = (x_r, y_r, z_r) \quad (6)$$

$$o_r = \psi_r \quad (7)$$

$$f_c = \begin{cases} \text{true} \\ \text{false} \end{cases} \quad \begin{array}{c} \text{if sUAV crashes,} \\ \text{otherwise.} \end{array} \quad (8)$$

$$f_b = \begin{cases} \text{true} \\ \text{false} \end{cases} \quad \begin{array}{c} \text{if } p_r > f_{ROI}, \\ \text{otherwise.} \end{array} \quad (9)$$

where $p_r$ is the position of the robot and $o_r$ is the orientation of the robot in the world Cartesian frame; $o_r$ is simplified to $\psi_r$ because multi-rotors primarily control their orientation based on their yaw angle only; $f_c$ is a discrete state that defines whether the sUAV has crashed with an obstacle and $f_b$ determines whether the sUAV is flying beyond the limits of the flying area.

The victim states are defined by Equation 10,

$$S_v = (p_v, o_v, f_v) \quad (10)$$

$$p_v = (x_v, y_v, z_v) \quad (11)$$

$$o_v = \psi_v \quad (12)$$

$$f_v = \begin{cases} \text{true} \\ \text{false} \end{cases} \quad \begin{array}{c} \text{if target is found,} \\ \text{otherwise.} \end{array} \quad (13)$$

where $p_v$ is the position of the victim and $o_v$ is the orientation of the victim in the world Cartesian frame; The orientation of a person can likewise be simplified using $\psi_v$ only, the victim's yaw angle; $f_v$ is the discrete state of whether the target has been found by the sUAV.

**Table 1**: **sUAV set of actions**

| **Action** | $\mathbf{p_r(m)}$ | $\mathbf{o_r(rad)}$ |
|---|---|---|
| Forward | $(0.3, 0, 0)$ | $0$ |
| Right | $(0, 0, 0)$ | $\pi/4$ |
| Left | $(0, 0, 0)$ | $-\pi/4$ |
| Up | $(0, 0, 0.3)$ | $0$ |
| Down | $(0, 0, -0.3)$ | $0$ |
| Hover | $(0, 0, 0)$ | $0$ |

*Actions*

The multi-rotor interacts with the environment using a set of six actions, as shown in Table 1.

For any flight mission, the POMDP assumes that the sUAV is already flying when it starts its autonomous interaction with the environment. Therefore, other common sUAV actions such as autonomous takeoff, landing and return home are addressed by the sUAV autopilot instead.

*Transition Function*

The motion model of the sUAV is based on the set of actions described above. Changes in rotation can be modelled using the rotation matrix $\mathbf{R}$ of a quad-rotor [32]. Owing to the kinematics of a multi-rotor sUAV, the evaluation of its $\theta$ and $\phi$ angles are discarded in the model. Additionally, an angle deviation $\varphi$ is added to $\psi$ to incorporate uncertainty caused by pose estimation errors. This uncertainty is modelled as a normal distribution with mean $\mu = \psi_r$ and standard deviation $\sigma = 3.0°$. Thus, $\mathbf{R}$ is simplified as shown in Equation 14.

$$\mathbf{R_r} = \begin{bmatrix} \cos(\psi_r + \varphi_r) & -\sin(\psi_r + \varphi_r) & 0 \\ \sin(\psi_r + \varphi_r) & \cos(\psi_r + \varphi_r) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{14}$$

The transformation matrix to model changes in position per time step is defined through Equation 15,

$$p_{r_{t+1}} = p_{r_t} + \mathbf{R_{r_t}} \mathbf{\Delta p_{r_t}} \tag{15}$$

which can be expanded as:

$$\begin{bmatrix} x_{r_{t+1}} \\ y_{r_{t+1}} \\ z_{r_{t+1}} \end{bmatrix} = \begin{bmatrix} x_{r_t} \\ y_{r_t} \\ z_{r_t} \end{bmatrix} + \begin{bmatrix} \cos(\psi_{r_t} + \varphi_{r_t}) & -\sin(\psi_{r_t} + \varphi_{r_t}) & 0 \\ \sin(\psi_{r_t} + \varphi_{r_t}) & \cos(\psi_{r_t} + \varphi_{r_t}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \Delta x_{r_t} \\ \Delta y_{r_t} \\ \Delta z_{r_t} \end{bmatrix} \tag{16}$$

where $\Delta p_{r_t} = (\Delta x_{r_t}, \Delta y_{r_t}, \Delta z_{r_t})$ is the change in the robot's location from time step $t$ to time step $t + 1$.

The dynamics of the sUAV through changes in position ($\Delta p_{r_t}$) are modelled using a system identification process. Further details on the calculation of $\Delta p_{r_t}$ can be found in the Appendix.

*Rewards*

The system rewards $R$ is defined by Equation 17,

$$R = r_{\text{move}} + r_{\text{crash}} + r_{\text{out}} + r_f + r_d \tag{17}$$

where $r_{\text{move}}$ is the cost (negative reward) per move, which encourages the sUAV to find the victim in a minimum number of steps. $r_{\text{crash}}$ is the cost for the drone by crashing itself with an obstacle; $r_{\text{out}}$ is the cost of flying beyond the explicitly defined Region of Interest (ROI) limits; $r_f$ is the reward if the victim is found; and $r_d$ is the cost given by the Euclidean distance between the sUAV and victim locations, as defined in Equation 18:

$$r_d = -\sqrt{(p_{r_x} - p_{v_x})^2 + (p_{r_y} - p_{v_y})^2 + (p_{r_z} - p_{v_z})^2} \tag{18}$$

The cost values of $r_d$ are directly proportional to the distance between the robot and the victim in the $x, y, z$ axes. Including $r_d$ encourages the sUAV to get closer to the victim and collect better image representations from vision-based sensors. The values for the rest of the rewards were acquired empirically, as shown in Table 2.

**Table 2**: **System rewards for a sUAV target finding task**

| **Reward** | **Value** |
|---|---|
| $r_{\text{move}}$ | $-10$ |
| $r_{\text{crash}}$ | $-150$ |
| $r_{\text{out}}$ | $-300$ |
| $r_f$ | $500$ |

*Observation Space*

The set of observations $O$ for this problem are defined as:

$$O = (o_{p_r}, o_{p_v}, o_{v_f}) \tag{19}$$

where $o_{p_r}$ is the local pose estimation of the robot; $o_{p_v}$ is the local pose estimation of the victim; and $o_{v_f}$ is a discrete observation defining whether the victim has been detected by the sUAV computer vision object detector. The victim pose observation is received only when the person is found. The object detector executes a pre-trained off-the-shelf deep learning object detector model. The datasets used to fit these models commonly contain thousands of images of classes collected from a frontward-looking camera configuration.

*Observation Model*

The observation model $\mathcal{Z}$ comprises the estimated sUAV position in the world coordinate frame and the location of the victim if it is detected by the downward-looking camera. The detection of a victim relies on the camera's Field of View (FOV). The modelling of the FOV depends on the sensor properties, the robot's pose and heading observations. First, the horizontal and vertical FOV angles are calculated as defined in Equation 20 and Equation 21.

$$\text{FOV}_\text{V} = 2\tan^{-1}\left(\frac{w}{2f}\right) \tag{20}$$

$$\text{FOV}_\text{H} = 2\tan^{-1}\left(\frac{h}{2f}\right) \tag{21}$$

where $w$ is the sensor width, $h$ is the sensor height, and $f$ is the focal length of the downward-looking camera. The extent of the observed FOV area (or footprint) is calculated as:

$$l_{\text{top}} = p_{r(z)} \cdot \tan(\alpha + 0.5 \cdot \text{FOV}_{\text{H}}) \tag{22}$$

$$l_{\text{bottom}} = p_{r(z)} \cdot \tan(\alpha - 0.5 \cdot \text{FOV}_{\text{H}}) \tag{23}$$

$$l_{\text{left}} = p_{r(z)} \cdot \tan(\alpha + 0.5 \cdot \text{FOV}_{\text{V}}) \tag{24}$$

$$l_{\text{right}} = p_{r(z)} \cdot \tan(\alpha - 0.5 \cdot \text{FOV}_{\text{V}}) \tag{25}$$

where $l_*$ is the footprint extent of any collected image and $\alpha$ is the gimbal angle of the camera from the vertical (i.e. 0 degrees), as depicted in Figure 2.

The calculation of the footprint point coordinates with its center in the origin is defined as:

$$c_1 = (l_{\text{top}}, l_{\text{left}}, 0) \tag{26}$$

$$c_2 = (l_{\text{top}}, l_{\text{right}}, 0) \tag{27}$$

$$c_3 = (l_{\text{bottom}}, l_{\text{right}}, 0) \tag{28}$$

$$c_4 = (l_{\text{bottom}}, l_{\text{left}}, 0) \tag{29}$$

A transformation matrix is then calculated to locate the point coordinates within the sUAV reference frame:

$$\begin{bmatrix} c'_x \\ c'_y \\ c'_z \end{bmatrix} = \begin{bmatrix} p_{r(x)} \\ p_{r(y)} \\ p_{r(z)} \end{bmatrix} + \begin{bmatrix} \cos(\psi_r) & -\sin(\psi_r) & 0 \\ \sin(\psi_r) & \cos(\psi_r) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_x \\ c_y \\ c_z \end{bmatrix} \tag{30}$$

A victim is predicted to be within the camera's FOV if a belief location point of the person is positioned inside the rectangular polygon from the group of $c$ points. This calculation is performed as the sum of the angles between the victim belief position point and each pair of points that comprise the rectangle [33], as defined in Equation 31.

$$\theta = \sum_{i=1}^{4} \left\{ \tan^{-1}\left[\frac{c_{i+1}(y) - p_t(y)}{c_{i+1}(x) - p_t(x)}\right] - \tan^{-1}\left[\frac{c_i(y) - p_t(y)}{c_i(x) - p_t(x)}\right] \right\} \tag{31}$$

From this formulation, the victim location point is predicted to be inside of the camera's FOV if $\theta = 2\pi$. Perfect accuracy is, however, assumed here from any vision-based model implemented in the detection subsystem. Target detection uncertainty from computer vision, and specifically, pre-trained deep learning detectors, occurs from different factors, including image noise, illumination conditions, image resolution, image representations of people from the dataset and camera configuration. Even though these object detection models can be improved using different techniques, some of these factors (that cause uncertainty) can be simulated by extending the target finding modelling. Taking into account that off-the-shelf object detection models give their best results when achieving close image representations from their trained datasets (e.g. ImageNet, COCO), a positive person detection is simulated if the sUAV and victim heading angles are similar (*i.e.* aligned) each other, as defined in Equation 32.

$$\text{target} = \begin{cases} \text{found} & \text{if } \theta = 2\pi \text{ and } |\psi_r - \psi_t| < 30^\circ \\ \text{not found} & \text{otherwise} \end{cases} \tag{32}$$

From the observation model described above and the reward function defined in Equation 17 and Equation 18, target detection uncertainty from deep learning object detectors is expected to be reduced by encouraging the robot to fly at a close distance between the victim and the drone itself. Furthermore, the sUAV will also adjust its orientation to match potentially a scene representation that increases the likelihood of the detector to detect a person.

## 5. IMPLEMENTATION

*System Architecture*

The proposed framework comprises four modules that interact with a multi-rotor sUAV under simulation, as shown in Figure 3.

*Software in the Loop module*—The sUAV is controlled using PX4, an open source software developed by the Dronecode Project [34]. The PX4 architecture is comprised of two layers: the flight stack layer and the middleware layer. The flight stack layer contains a pipeline of flight controllers for a rich set of UAVs (multi-rotors, fixed-wing and VTOL) and altitude and position estimators. These estimators usually make predictions from one or multiple sensor inputs such as IMU and GPS. The PX4 flight controller follows a feedback control loop process for position and velocity set-point values, a PID controller, and feedback signals from the estimators. The middleware layer contains the device drivers for the sUAV sensors, communication interfaces, and a simulation layer.

The simulation layer (PX4 Software in the Loop (SITL)) ports the PX4 architecture in a simulated sUAV platform and environment to a local machine. The sUAV (a 3DR Iris) and the download-looking camera (an ov7251) are simulated under the Gazebo simulator.

*Hardware in the Loop module*—Given hardware limitations on sUAVs, an Intel® Neural Compute Stick is integrated into the system. The stick is a plug and play Vision Processing Unit (VPU) device, an optimised microprocessor that boosts inference from deep learning models. Intel provides a specific software toolkit to obtain performance gains through the OpenVINO library. OpenVINO supports a range of deep learning frameworks such as TensorFlow, Caffee and PyTorch, and optimised versions of OpenCV and OpenVX for standard image processing operations. The source code for target detection with OpenVINO contains ROS bindings in Python, allowing the use of this hardware for both real tests and simulation environments via PX4 SITL and Gazebo. The latter provides, therefore, support under a Hardware in the Loop (HITL) interface.

*Detection module*—The detection module comprises a deep learning object detector. The selected detector is an open-

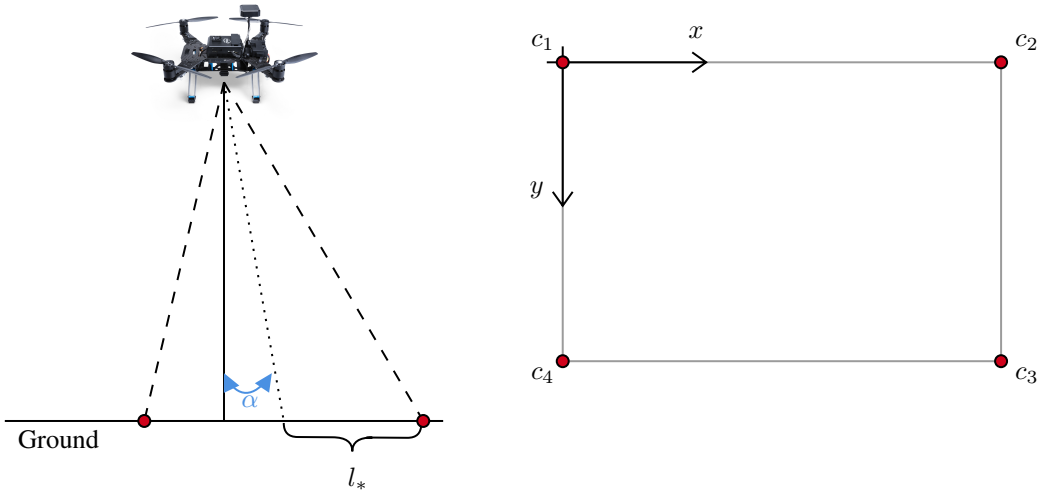



**Figure 2**: **FOV and 2D image representation from a vision-based camera pointing to the ground with an angle $\alpha$ from the vertical.**

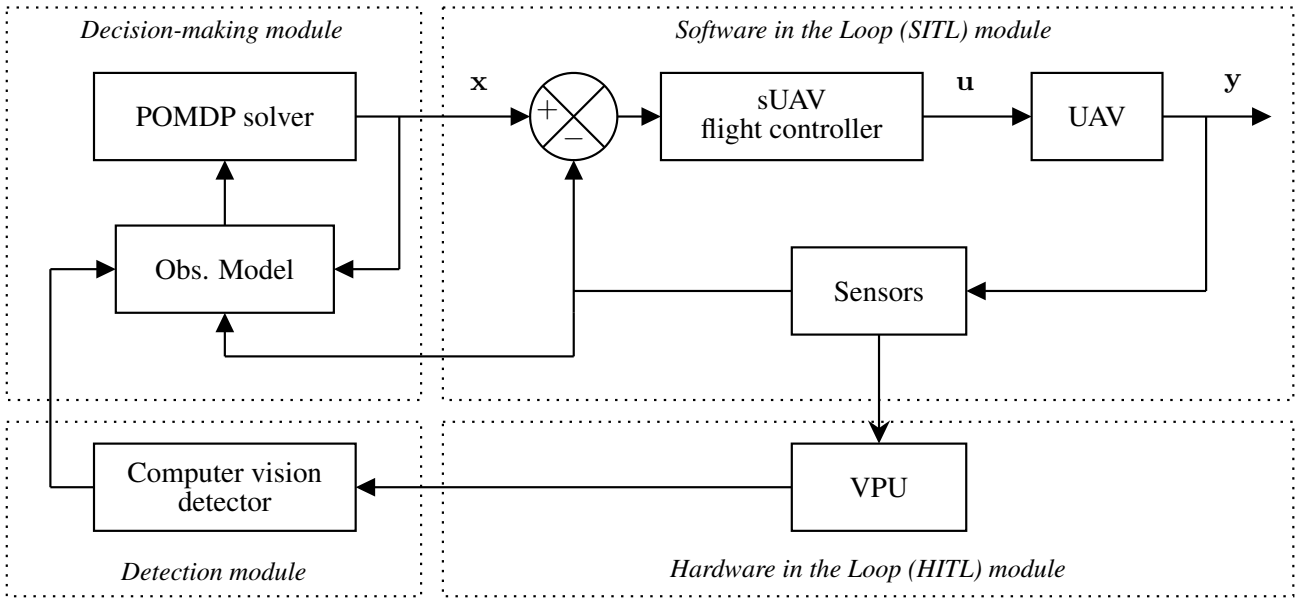



**Figure 3**: **Proposed system architecture under a simulation environment.**

source instance of the Google MobileNet Single Shot Multi-box Detector (SSD) architecture [35]. The model is deployed in caffe and fit using pre-trained weights from the PASCAL VOC0712 dataset, achieving a mean average precision of 72.7%. For every read frame, the detection module subscribes to a downward-looking camera ROS topic included in the Iris model. Input frames are resized into dimensions of $300 \times 300$. Any object detections with a confidence value greater than 30% from the output layer are displayed in the processed frame. If the chosen class (*i.e.* the object of interest) is displayed, the position of the object is estimated following the formulation described in Section 4. An illustration of a person detection from the deep learning model is shown in Figure 4.

*Decision-making module*—The POMDP formulation, which was described in Section 4, is computed using the TAPIR toolkit [36]. The toolkit incorporates the online ABT solver which handles continuous states. Additionally, TAPIR includes a ROS interface that eases communication between the online solver and the sUAV flight controller (PX4 SITL).

*Experimentation Setup*

The system was tested in a cluttered and GPS-denied simulated environment using the Gazebo simulator. The goal of the sUAV is to find a child who needs assistance to contextualise the experiments under SAR situations. The flying area, with dimensions $6 \times 6$ $m$ in length and width and 3 $m$ in height, contains several column obstacles using cardboard boxes and a table, as shown in Figure 1.

The test scenario contains three cardboard obstacles in form of columns placed throughout the scene, a table that partially occludes the victim, a safety net which delimits the flying area and a child dummy to be found. For all the experiments, the child is always located under the table at world Cartesian

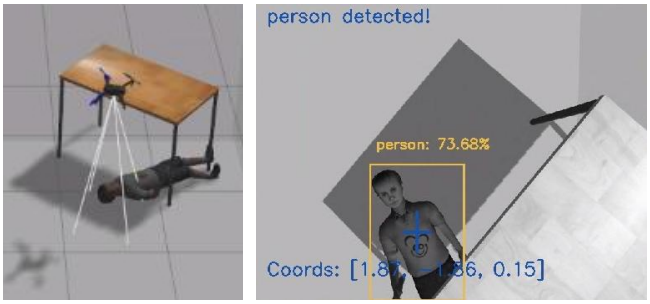



**Figure 4**: Detection example of a child dummy from a downward-looking camera under Gazebo SITL. the sUAV takes actions to align itself with the child and minimise uncertainty.

coordinates $(2.0, -2.0, 0.2)$. Visuals of the child come from a downward-looking camera attached to the Iris UAV frame. The sUAV uses the take-off and landing modes from the PX4 autopilot, and start the navigation task at world Cartesian coordinates $(-2.0, -2.0, 2.0)$.

## 6. RESULTS

The autonomous decision-making system and target detection uncertainty is evaluated with two types of setups. The first setup declares different levels of child location uncertainty within the flying area. The second setup evaluates the system under different child orientation configurations.

*Location Uncertainty*

Location uncertainty for the child is defined through three case studies, illustrate in Figure 5 and described as follows:

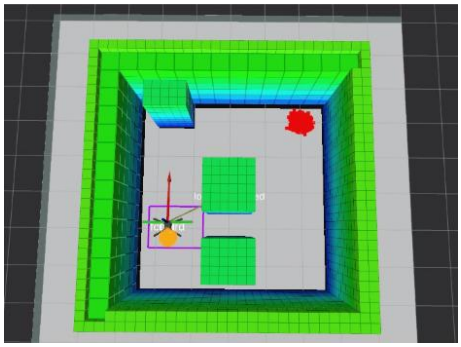
A *single location estimation*: the belief states of the victim location are represented as a point cloud, sorted as a normal distribution ($\mu = 0m, \sigma = 0.5m$) at one specific region (top right corner from Figure 5a).
B *multiple location estimation*: the belief states of victim location now include two possible locations, which are sorted as a normal distribution ($\mu = 0m, \sigma = 0.5m$) and represented as two point clouds (Figure 5b).
C *uniform location estimation*: the belief states of the victim location are uniformly distributed into the flying area, assuming thus, that the location of the victim is unknown (Figure 5c).

Each case study was run $40$ times, with a mean duration per time step of $1.049$ seconds. The success rate per case study is illustrated in Table 3.

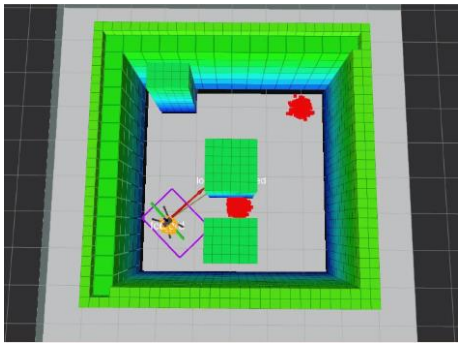
**Table 3**: **Location uncertainty success rates under SITL.**

| Case Study | Success Rate (Target found) | Failure Rate (Crash) | Failure Rate (Timeout) |
|---|---|---|---|
| A | 100% | 0% | 0% |
| B | 87.5% | 12.5% | 0% |
| C | 52.5% | 2.5% | 45% |

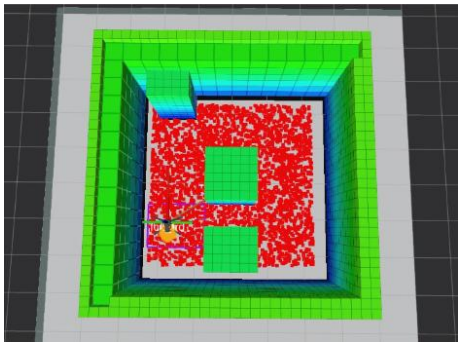
Overall, the success rate for case study *A* is remarkable, achieving $100\%$ of successful runs. The sUAV was able to follow the reward structure with the absence of crash reports or flying beyond the ROI limits. The drone planned a trajectory that ensured matching its location above the target



(a)



(b)



(c)

**Figure 5**: **SITL environment illustration that overlays an occupancy map (in green) and the belief map (in red).**

point cloud first, followed by adjusting its heading angle $\psi$ to filter target location particles until the detection subsystem found the target. Similarly, case study *B*, which encouraged the sUAV to fly nearby two of the cardboard boxes, achieved a success rate of $87.5\%$. As shown in Figure 5b, a second point cloud of child location estimations was added in the middle of two cardboard columns. Even though the drone was able to adjust its trajectory after filtering all the particles in zones without the presence of the child, drifts in the $x$ and $y$ axes were clearly visible while the sUAV spun next to the columns. For case study *C*, the number of failures became evident by exceeding the maximum flight time of the sUAV. The evaluation of the sUAV and target heading angles in the observation model provoked an increase in the number of time steps to discard particles, as demonstrated in Figure 6.

The number of time steps was consistent for case study *A*, with a median of $35$ steps and an inter-quartile range of four steps. When the number of target particles covers a bigger

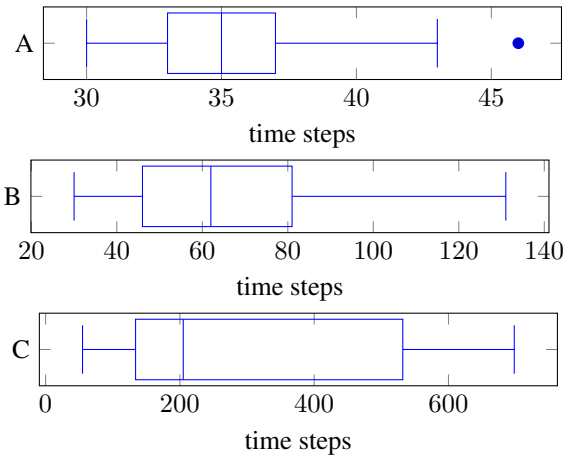
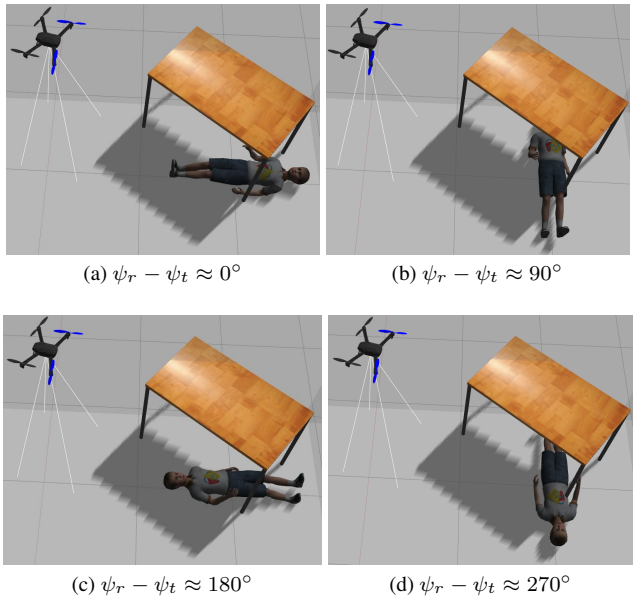



**Figure 6**: **Time steps of successful runs under location uncertainty.**

region in the flying area, so is the number of required actions (spins) to evaluate different heading angles. In fact, the median number of time steps almost doubled for case study *B* and a evident greater variability on time steps for case study *C*.

*Orientation Uncertainty*

The sUAV navigation task and the POMDP observation model were also evaluated by varying the heading (or orientation) of the child. The child was placed using four orientation configurations, as shown in Figure 7. For each configuration, the distribution of possible locations (location uncertainty) for the dummy followed the settings of case study *A*.



(a) $\psi_r - \psi_t \approx 0°$ (b) $\psi_r - \psi_t \approx 90°$

(c) $\psi_r - \psi_t \approx 180°$ (d) $\psi_r - \psi_t \approx 270°$

**Figure 7**: **Dummy child at different orientation configurations.**

The success rate per case study is illustrated in Table 4.

The sUAV was able to navigate and detect the child with a mean success rate of 98%. Indeed, the lost child was always found when the particle distribution surrounded the child location at different orientations. The distribution of time steps for all the episodes is illustrated in Figure 8.

**Table 4**: **Success rates for defined levels of target uncertainty under SITL.**

| Case Study | Success Rate (Target found) | Failure Rate (Crash) | Failure Rate (Timeout) |
|---|---|---|---|
| A | 100% | 0% | 0% |
| B | 97.5% | 2.5% | 0% |
| C | 100% | 0% | 0% |
| D | 95% | 5.0% | 0% |

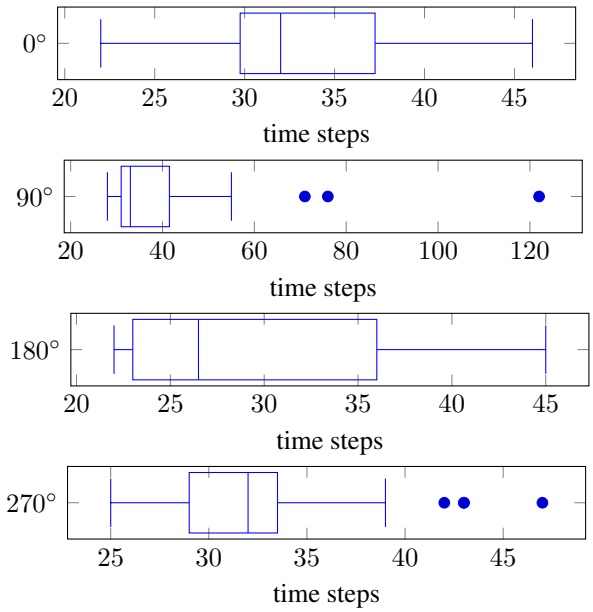



**Figure 8**: **Time steps of successful runs under orientation uncertainty.**

The median number of time steps for the sUAV while collecting child frames was the lowest when its image representations were aligned with the sUAV ($\psi_r - \psi_t \approx 0°$), as depicted in Figure 4. Conversely, an alignment difference of 180° resulted in a bigger number of time steps to detect the child. The big variance and outliers in time steps for alignment differences of 90° and 270° happened because of the partial occlusion from the table, requiring a higher number of actions for the sUAV to get a clear visual of the child. This variance also illustrates the adaptability of the sUAV in a environment under uncertainty, given the way PX4 Gazebo SITL emulates stochastic odometry errors in the drone and external disturbances in the environment.

## 7. Conclusions

This paper introduced a solution to the problem of searching victims with a prior probability with respect to the likely location of the victims. The problem was formulated as a Partially Observable Markov Decision Process to solve the task as a multi-objective sUAV navigation problem that incorporates uncertainty and partial observability represented in state belief. The proposed framework was validated on SAR mission simulated in an indoor cluttered environment under different levels of location and orientation uncertainties. The

experiments were tested using PX4, Gazebo SITL, ROS, an Intel VPU for deep learning inference (HITL) and TAPIR.

The problem formulation and system architecture constitute a substantial extension on previous contributions on autonomous sUAV decision-making for target finding under uncertainty. Specifically, the proposed system presents an extension on the work of Vanegas et al. [37] by incorporating:

- An observation model that incorporates target detection uncertainty for vision-based object detectors.
- Experimentation close to real-world conditions under SITL and HITL capabilities.
- Onboard target detection using deep learning and VPUs for real-time inference under resource-constrained hardware.

The results demonstrate the capability of the proposed system to deal with high levels of environment and target detection uncertainty as well as progress towards automating surveillance operations for applications that require rapid intervention such as SAR.

## APPENDIX

*System Identification*

Motion response of the plant (*i.e.* sUAV) is collected by measuring the robot's position values $y(t)$ under a step response $r(t)$ in $x$, $z$ and $\psi$ from the world coordinate frame. As an illustration, the process to identify the system under a step position response in $x$ is illustrated herewith. First, the transfer function of the plant was calculated using the System Identification Toolbox™ from MATLAB®, as shown in Equation 33.

$$F(s) = \frac{0.204s + 1.136}{s^2 + 1.253s + 1.134} \tag{33}$$

The calculation of the plant in time discrete is done using the Tustin approximation method, which is defined in Equation 34,

$$s \approx \frac{2(z-1)}{T_s(z+1)} \tag{34}$$

where $T_s$ is the sampling period. Assuming $T_s = 0.1s$, the discretised plant $F(z)$ equals:

$$F(z) = \frac{0.01224 + 0.005333z^{-1} - 0.006905z^{-2}}{1 - 1.872z^{-1} + 0.8824z^{-2}} \tag{35}$$

The difference equation from $F(z)$ is calculated by applying the inverse Z transform:

$$F(z) = \frac{Y(z)}{R(z)}$$

$$Y(z) = \frac{\left(A_0 + A_1z^{-1} + A_2z^{-2}\right)R(z)}{1 + B_1z^{-1} + B_2z^{-2}}$$

$$\begin{aligned} y(k) = A_0r(k) + A_1r(k-1) + A_2r(k-2) \\ -B_1y(k-1) - B_2y(k-2) \end{aligned} \tag{36}$$

where

$$A_0 = 0.012237830217107$$

$$A_1 = 0.005333276901521$$

$$A_2 = -0.006904553315587$$

$$B_1 = -1.871779712793530$$

$$B_2 = 0.882425299507294$$

The value of $\Delta x_{r_t}$ is ultimately calculated by iterating Equation 36 every $T_s$ seconds until reaching the total duration per time step.

## ACKNOWLEDGMENTS

The authors would like to thank The Commonwealth Scientific and Industrial Research Organisation (CSIRO) through the CSIRO Data61 PhD and Top Up Scholarships (Agreement 50061686), and the Australian Research Council (ARC) through the ARC Discovery Project 2018 'Navigating under the forest canopy and in the urban jungle' for funding this research work.

## REFERENCES

[1] G. Pajares, "Overview and current status of remote sensing applications based on unmanned aerial vehicles (uavs)," *Photogrammetric Engineering & Remote Sensing*, vol. 81, no. 4, pp. 281–330, Apr. 2015.

[2] B. J. Stark and Y. Q. Chen, "Remote sensing methodology for unmanned aerial systems," in *Unmanned Aircraft Systems*, E. M. Atkins, A. Ollero, and A. Tsourdos, Eds. New York, USA: Wiley, 2016, ch. 2, pp. 17–27.

[3] K. Dalamagkidis, K. P. Valavanis, and L. A. Piegl, *On integrating unmanned aircraft systems into the national airspace system: Issues, challenges, operational restrictions, certification, and recommendations*. Dordrecht: Springer Netherlands, 2012.

[4] A. Carrio, C. Sampedro, A. Rodriguez-Ramos, and P. Campoy, "A review of deep learning methods and applications for unmanned aerial vehicles," *Journal of Sensors*, vol. 2017, pp. 1–13, 2017.

[5] M. Bouhali, F. Shamani, Z. E. Dahmane, A. Belaidi, and J. Nurmi, "Fpga applications in unmanned aerial vehicles - a review," in *Lecture Notes in Computer Science*, 2017, vol. 10216 LNCS, pp. 217–228.

[6] Y. Zhao, Z. Zheng, and Y. Liu, "Survey on computational-intelligence-based uav path planning," *Knowledge-Based Systems*, vol. 158, pp. 54–64, Oct. 2018.

[7] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, MA: MIT Press, 2005.

[8] Australian Bureau of Statistics, "Understanding natural hazard impacts on australia," Canberra, 2008. [Online]. Available: https://www.abs.gov.au/ausstats/abs@.nsf/Previousproducts/1301.0FeatureArticle42008?

opendocument{&}tabname=Summary{&}prodno=1301.0{&}issue=2008{&}num={&}view=

[9] ——, "Natural disasters in australia," Canberra, 2008. [Online]. Available: https://www.abs.gov.au/ausstats/abs@.nsf/7d12b0f6763c78caca257061001cc588/fecb2ab6de16171eca2570de0005871b!OpenDocument

[10] R. Z. B. Bravo, A. Leiras, and F. L. Cyrino Oliveira, "The use of uavs in humanitarian relief: An application of pomdp-based methodology for finding victims," *Production and Operations Management*, vol. 28, no. 2, pp. 421–440, Feb. 2019.

[11] M. Erdelj and E. Natalizio, "Uav-assisted disaster management: Applications and open issues," in *International Conference on Computing, Networking and Communications*. IEEE, Feb. 2016, pp. 1–5.

[12] R. Murphy, "Human–robot interaction in rescue robotics," *Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, vol. 34, no. 2, pp. 138–153, May 2004.

[13] C. Sampedro, A. Rodriguez-Ramos, H. Bavle, A. Carrio, P. de la Puente, and P. Campoy, "A fully-autonomous aerial robot for search and rescue applications in indoor environments using learning-based techniques," *Journal of Intelligent & Robotic Systems*, pp. 1–27, Jul. 2018.

[14] T. Tomic, K. Schmid, P. Lutz, A. Domel, M. Kassecker, E. Mair, I. Grixa, F. Ruess, M. Suppa, and D. Burschka, "Toward a fully autonomous uav: Research platform for indoor and outdoor urban search and rescue," *IEEE Robotics & Automation Magazine*, vol. 19, no. 3, pp. 46–56, Sep. 2012.

[15] V. San Juan, M. Santos, and J. M. Andújar, "Intelligent uav map generation and discrete path planning for search and rescue operations," *Complexity*, vol. 2018, pp. 1–17, 2018.

[16] M. J. Kochenderfer, *Decision making under uncertainty: theory and application*. Cambridge, MA: MIT Press, 2015.

[17] S. A. Miller, Z. A. Harris, and E. K. Chong, "A pomdp framework for coordinated guidance of autonomous uavs for multitarget tracking," *EURASIP Journal on Advances in Signal Processing*, vol. 2009, no. 1, p. 724597, Dec. 2009.

[18] S. Ragi and E. K. P. Chong, "Uav path planning in a dynamic environment via partially observable markov decision process," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 49, no. 4, pp. 2397–2412, Oct. 2013.

[19] Y. Zhao, X. Wang, W. Kong, L. Shen, and S. Jia, "Decision-making of uav for tracking moving target via information geometry," in *Chinese Control Conference*. Chengdu, China: IEEE, Jul. 2016, pp. 5611–5617.

[20] C. M. Eaton, E. K. Chong, and A. A. Maciejewski, "Robust uav path planning using pomdp with limited fov sensor," in *Conference on Control Technology and Applications (CCTA)*. Hawaii, USA: IEEE, Aug. 2017, pp. 1530–1535.

[21] F. Vanegas and F. Gonzalez, "Enabling uav navigation with sensor and environmental uncertainty in cluttered and gps-denied environments," *Sensors*, vol. 16, no. 5, p. 666, May 2016.

[22] D. Silver and J. Veness, "Monte-carlo planning in large pomdps," *Neural Information Processing Systems*, pp. 1–9, 2010.

[23] H. Kurniawati and V. Yadav, "An online pomdp solver for uncertainty planning in dynamic environment," in *Springer Tracts in Advanced Robotics*, 2016, vol. 114, pp. 611–629.

[24] K. P. Valavanis and G. J. Vachtsevanos, "Future of unmanned aviation," in *Handbook of Unmanned Aerial Vehicles*, K. P. Valavanis and G. J. Vachtsevanos, Eds. Dordrecht: Springer Netherlands, 2015, ch. 126, pp. 2993–3009.

[25] S. Ragi and E. K. P. Chong, "Uav guidance algorithms via partially observable markov decision processes," in *Handbook of Unmanned Aerial Vehicles*, K. Valavanis and G. Vachtsevanos, Eds. Dordrecht: Springer Netherlands, 2015, ch. 73, pp. 1775–1810.

[26] C. Ponzoni Carvalho Chanel, A. Albore, J. T'Hooft, C. Lesire, and F. Teichteil-Königsbuch, "Ample: an anytime planning and execution framework for dynamic and uncertain problems in robotics," *Autonomous Robots*, vol. 43, no. 1, pp. 37–62, Jan. 2019.

[27] M. Chen, E. Frazzoli, D. Hsu, and W. S. Lee, "Pomdp-lite for robust robot planning under uncertainty," *International Conference on Robotics and Automation*, pp. 5427–5433, Feb. 2016.

[28] U. Ilhan, L. Gardashova, and K. Kilic, "Uav using decpomdp model for increasing the level of security in the company," *Procedia Computer Science*, vol. 102, pp. 458–464, 2016.

[29] S. C. W. Ong, S. W. Png, D. Hsu, and W. S. Lee, "Pomdps for robotic tasks with mixed observability," in *Robotics: Science and Systems V*. Seattle, US: Robotics: Science and Systems Foundation, Jun. 2009.

[30] C. Chanel, F. Teichteil-Königsbuch, and C. Lesire, "Multi-target detection and recognition by uavs using online pomdps," in *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*. Bellevue, Washington: AAAI Press, 2013, pp. 1381–1387.

[31] A. Al-Kaff, D. Martín, F. García, A. de la Escalera, and J. María Armingol, "Survey of computer vision algorithms and applications for unmanned aerial vehicles," *Expert Systems with Applications*, vol. 92, pp. 447–463, Feb. 2018.

[32] A. Chovancová, T. Fico, L. Chovanec, and P. Hubinsk, "Mathematical modelling and parameter identification of quadrotor (a survey)," *Procedia Engineering*, vol. 96, pp. 172–181, 2014.

[33] P. Bourke, "Polygons and meshes," 1997. [Online]. Available: http://paulbourke.net/geometry/polygonmesh/

[34] L. Meier, D. Honegger, and M. Pollefeys, "Px4: A node-based multithreaded open source robotics framework for deeply embedded platforms," in *International Conference on Robotics and Automation*, no. June. IEEE, May 2015, pp. 6235–6240.

[35] Chuanqi305, "Caffe implementation of google mobilenet ssd detection network, with pretrained weights on voc0712 and map=0.727." 2019. [Online]. Available: https://github.com/chuanqi305/MobileNet-SSD

[36] D. Klimenko, J. Song, and H. Kurniawati, "Tapir: A software toolkit for approximating and adapting pomdp solutions online," in *Australasian Conference on*

*Robotics and Automation*, Melbourne, Australia, 2014, pp. 1–9.

[37] F. Vanegas, D. Campbell, M. Eich, and F. Gonzalez, "Uav based target finding and tracking in gps-denied and cluttered environments," in *International Conference on Intelligent Robots and Systems*. Daejeon, South Korea: IEEE/RSJ, Oct. 2016, pp. 2307–2313.

## BIOGRAPHY

***Juan Sandino*** *holds a BEng (Mechatronics) and is currently undertaking a PhD in robotics and autonomous systems at QUT, Australia. His primary interests comprise autonomous UAV decision-making, machine learning and computer vision for UAV remote sensing, with a focus on hyperspectral and high-resolution image processing. Juan has worked for research projects in biosecurity, environment monitoring and search and rescue.*

***Fernando Vanegas*** *received his B.S. in Mechatronics Engineering from UMNG in 2004 and M.Sc. in Electrical Engineering from Halmstad University in 2008. He is currently a Ph.D. candidate in Robotics and Autonomous Systems at The Australian Research Centre for Aerospace Automation and Queensland University of Technology. His current research activities include motion planning for UAV in cluttered and uncertain environments modelled as POMDPs.*

***Felipe Gonzalez*** *holds a BEng (Mech) and a PhD from the University of Sydney. Gonzalez is an Associate Professor at the School of Electrical Engineering and Robotics (ECCS), Science and Engineering Faculty with a passion for innovation in the fields of aerial robotics and automation. Gonzalez interest is in creating aerial robots, drones or UAVs that possess a high level of cognition using efficient on-board computer algorithms using advanced optimisation and game theory approaches that assist us to understand and improve our physical and natural world. Dr Gonzalez lead the Airborne Sensing Lab at QUT.*

***Frederic Maire*** *received the M.Sc. degree in pure mathematics and computer science engineering in 1989 and the Ph.D. degree in discrete mathematics from the Universite Pierre et Marie Curie, Paris 6, France, in 1993. He is currently a Senior Lecturer with the School of Electrical Engineering and Robotics, Queensland University of Technology, Brisbane, Australia. His research interests include computer vision and robotics.*