UAV Based Target Finding and Tracking in GPS-Denied and Cluttered Environments

Fernando Vanegas[1], Duncan Campbell[2], Markus Eich[3], Felipe Gonzalez[4]

***Abstract*— In this paper we describe and flight test a novel system architecture for low cost muti-rotor unmanned aerial vehicles (UAVs) for searching, tracking and following a ground target. The UAV uses only on-board sensors for localisation within a GPS-denied space with obstacles. This mission is formulated as a Partially Observable Markov Decision Process (POMDP) and uses a modular framework that runs on the Robotic Operating System (ROS). This system computes a policy for executing actions instead of way-points to navigate and avoid obstacles. Results indicate that the system is robust to overcome uncertainties in localisation of both, the aircraft and the target and avoids collisions with the obstacles.**

I. INTRODUCTION

In recent years there has been an interest in using UAVs for performing a variety of civilian applications that covers environmental monitoring, topography, crop surveys, surveillance, aerial photography and filming. There are certain scenarios in which there is no GPS or there are structures and/or objects in the airspace that must be avoided and the UAV has to rely in its on-board sensors to localise itself. Examples for such scenarios are urban search and rescue missions or marine vessel inspections, where UAVs can provide visual information about defects and corrosions [7]. This type of scenarios present a challenge due to the uncertainties that are inherent to the perception and motion systems used by the UAV. An additional challenge is presented when a target must be found and followed due to the uncertainty in the target's motion.

POMDPs have been proposed for UAV navigation. In [1] and [2] a system for detecting multiple targets using online POMDPs is presented. This work uses a fixed wing aircraft with full GPS-Waypoint navigation capacity and presents a POMDP formulation in which the state is discrete and is comprised of zones, height levels and car models (as targets).

There are also several works that propose a system to track targets but assume perfect sensing and localisation of the UAV and with no obstacles to avoid [3], [4], [5]. Other works propose alternatives to the target tracking problem, showing results in simulation [6], [8]. Real flight tests for these type of scenarios is still an area that needs further exploration.

*This work was supported by QUT, ARCAA and COLCIENCIAS Scholarship 529

[1]Fernando Vanegas, [2]Duncan Campbell, [4]Felipe Gonzalez are with the School of Electrical Engineering and Computer Science, Queensland University of Technology and Australian Research Centre for Aerospace Automation, Brisbane, Australia. [3]Markus Eich is with the ARC Centre of Excellence for Robotic Vision (CE140100016), Queensland University of Technology (QUT), Brisbane, Australia. fernando.vanegasalvarez@hdr.qut.edu.au, da.campbell@qut.edu.au, felipe.gonzalez@qut.edu.au, markus.eich@qut.edu.au

A target tracking system using a simulated two-dimensional maze in which a ground robot moves in order to search and follow a target is presented in [10] and [11]. The scenario is discrete, and assumes that there is no uncertainty in robot control and sensing. A target tracking system using UAVs is described in [12] and [13] where the UAV has to fly above a target that has to be pursued and is moving on the ground. There is no uncertainty in motion or in the localisation and orientation of the UAV since a VICON system with 20 cameras is used. On the other hand, our system addresses a more challenging scenario with a UAV flying in a three-dimensional GPS-denied space with obstacles and uncertainty in both, the motion and the perception systems, due to the characteristics of its low cost sensors. This presents the challenge the system having to cope with multiple source of uncertainty, a larger number of state variables and a continuous state space and observation space in the POMDP formulation.

Our POMDP formulation presents a continuous state representation for the position in 3D space and orientation of the quad-copter and a planar position and orientation for the target moving on the ground.

The system uses a POMDP solver algorithm for calculating a policy of actions that has to be updated after every iteration in order to account for uncertainties in UAV motion and sensing in an indoor environment with obstacles. Calculations to account for a large number of possible sequences of actions and states of the vehicle in the scenario is needed but they must be conducted within a limited step time which is highly dependant on the system dynamics.

The modular system is developed using the Robotic Operating System (ROS). The system consists of four modules; a node that is running the online POMDP solver, another ROS node in charge of controlling the motion of the aircraft through four decoupled PID controllers, one in each degree of freedom. A third node that is calculating the perception of the environment and an existing ROS Driver for the Parrot AR Drone Quadrotor [9]. We illustrate the framework using an online POMDP solver *Adaptive belief-state Tree* (ABT) [14] for calculating and updating the policy on-line.

The main contributions of this paper are:

1) A system that models a Target Finding and Tracking mission as a POMDP, using an on-line algorithm that outputs actions instead of waypoints, and only relies on on-board sensors for localisation of both, the UAV and the Target. The state space is continuous, and there is uncertainty in motion and perception systems as well as in the location and the target's motion.

2) A framework for implementation and testing of a target tracking mission with a UAV in real flight using a online POMDP algorithm

This paper is organised as follows; section II covers a description of POMDP and an online solver: ABT. Section III describes the system architecture, section IV describes the scenario and the problem formulation. Section V describes the results and section VI provides conclusions and future areas of research.

II. Background

A. POMDP

In real world aerial, ground and underwater scenarios, the robot perception is limited by the type of sensors and the environment in which the robot is moving. The perception of the robot is not completely accurate and consequently there is uncertainty and errors in the estimation and localisation of the robot. This limitation in the sensory systems of robots is also known as partial observability [15], [16], [17]. One possible way of dealing with this uncertainty are Markov Decision Processes (MDPs) and Partially Observable Markov Decision Processes (POMDPs).

MDPs can be used to model the sequential decision problem under uncertainties [18]. MDPs are used for robotic missions to generate a policy that allows the robot to decide what sequence of actions should be taken in order to maximise a return or cost function, taking into account the uncertainties in motion. Plain MDPs assume that the states are completely observable which is not the case for a robot that has limitations in perception.

Partially Observable Markov Decision Processes (POMDPs), on the other hand, incorporate the uncertainties in sensing and the partial observability of the agent in the environment [19], [20].

Formally a Partially Observable Markov Decision Process is a tuple that consists of the following elements $(S, A, O, T, Z, R, \gamma)$ where S represents the set of states in the environment, A stands for set of possible action the agent can execute, O is the set of observations, T is the transition function for the state after taking an action, Z is the distribution function describing the probability of observing o from state s after taking action a, R is the set of rewards for every state and γ is the discount factor. POMDPs rely on the concept of belief-state which is a probability distribution of the system over all the possible states in its state-space representation at a particular time. It is denoted by *belief-state* b.

In a POMDP the state of the agent can not be observed exactly, instead, the agent receives an observation $o \in O$ determined by the probability distribution Z. A policy $\pi : \mathcal{B} \to A$ allocates an action a to each belief-state $b \in \mathcal{B}$, which is the set of possible belief-states. Given the current *belief-state* b, the objective of a POMDP algorithm is to find an optimal policy that maximizes the accumulated return when following a sequence of actions suggested by the policy π. The accumulated *discounted return* is the sum of the discounted rewards after executing every action in the sequence from time t onwards $R_t = \sum_{k=t}^{\infty} \gamma^{k-t} r_k$, where r_k is the immediate reward received at particular time step t for taking action a_t. The *Value function* is the expected return from belief-state b when following policy π, $V^\pi(b) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t | b, \pi]$. An optimal policy for the POMDP is the one that maximizes the value function $\pi^*(b) = \arg\max_\pi V^\pi(b)$.

B. ABT

In order to test the target finding and tracking problem formulated as a POMDP we implemented in hardware and software one of the fastest on-line POMDP algorithms to the authors knowledge, ABT [14]. This algorithm has been tested previously in simulations but not on-board a UAV.

ABT is an online POMDP solver that uses Monte Carlo Simulations and a set of state particles to represent the belief-state. It generates a search tree to store the policy. The root of tree is a node containing the state particles representing the initial belief-state of the environment.

In ABT the policy is updated after receiving an observation. ABT does not clear the policy tree after receiving and matching an observation but instead improves the policy based on the observation received. Moreover, ABT provides a mechanism for accepting changes in the environment and adapts its policy accordingly. This algorithm was selected for the implementation of the Target Finding and Tracking mission based on its capacity to handle a large number of possible observations that result from a larger uncertainty in the target's location. ABT can model the observations using a dynamic approach in which the number of observations does not have to be fixed as a constant value for the whole episode as opposed to POMCP [21]. ABT allows to model the observations as objects containing different variables. This approach allows to program a function that decide which observations are equal based on the values of the variables that compose the observation. Thanks to this feature of ABT, the observations do not have to be discretised but instead a function should be implemented defining whether two observations are equal based on the value of their variables. The implementation of this function for the Target Finding and Tracking problem is explained in section IV-E

III. System Architecture

The system is divided into four modules, shown in Fig. 1, one module for running the online POMDP solver, another module for controlling forward $\boldsymbol{V_{r_f}}$ and lateral velocity $\boldsymbol{V_{r_l}}$, yaw Ψ_r and altitude z_r of the multi-rotor, one module for calculating the multi-rotor position in the scenario $\boldsymbol{P_r} = (x_r, y_r, z_r, \Psi_r)$ based on the forward and lateral velocity, heading angle and altitude, estimated from the on-board sensors and the target position $\boldsymbol{P_t} = (x_t, y_t, z_t, \Psi_t)$, estimated by the target's pose in image if it detected by a downward looking camera. A fourth module which is the Autonomy Lab driver for ROS [6], receives the commands for actuation in roll, pitch, and yaw angles as well as the altitude and sends the sensor readings upon request. All nodes running in parallel in different threads allows the system to have different update rates for each module and that permits to have different levels of controllers.

A. *Online POMDP Module*

The Online POMDP module executes an online POMDP solver algorithm, ABT in our example. This module implements the ABT algorithm in which the POMDP elements described in section IV are programmed using the object oriented programming interfaces that are available [22]. In this module the POMDP model and solver are initialised according to input parameters and an initial belief-state b_0. It first produces a policy based on the initial belief-state b_0 and outputs the action a that maximises and expected return. The action is then executed by the motion control module, while the perception module calculates independently the robot position and heading angle and the target pose in case it is detected by the onboard downward camera. Based on the perception module data an observation is created and received after the action is executed.

The online POMDP module updates the belief-state b, to match the obtained observation and replenishes particles until a time-out is reached. Based on the current belief-state b the POMDP solver calculates and updates the policy and outputs the subsequent action based on the updated policy. The process repeats itself until a terminal state is reached i.e. the target has been detected and followed for at least ten seconds.

Finding the right time-step duration for the implementation of the target finding and tracking problem as an online POMDP, depends on the dynamic capabilities of the UAV used and on the complexity of the scenario. We set the online POMDP solver with a frequency of 1 Hz, since we know how the motion control system will respond within that time-frame and the online planner provides good policies within that time.

On one hand, having long time-steps for the policy update reduces the complexity for calculating a solution since the length of the planning horizon is reduced compared to a shorter time-step. On the other hand, having short time-steps reduces the uncertainty in motion and perception since the observation updates are more frequent. The POMDP online planner iteration time chosen balances between having an appropriate horizon and obtaining a good policy for the POMDP formulation.

B. *Motion Control Module*

This module is implemented as a ROS node that executes at a rate of 100 Hz, it has four PID controllers, one for each of the following states: Forward velocity $\boldsymbol{V_{r_f}}$, lateral velocity $\boldsymbol{V_{r_l}}$, Yaw angle Ψ_r and altitude z_r. The actions in the formulated POMDP are commands that set the reference values for each of the controllers. Each PID controller is tuned to obtain a fast response according to the required reference that is set by the action (see table I).

The output of the PID controllers (Table I) are pitch $\dot{\theta}$, roll $\dot{\phi}$, yaw $\dot{\psi}$ and $\dot{z}$ rates, that are sent to the AR

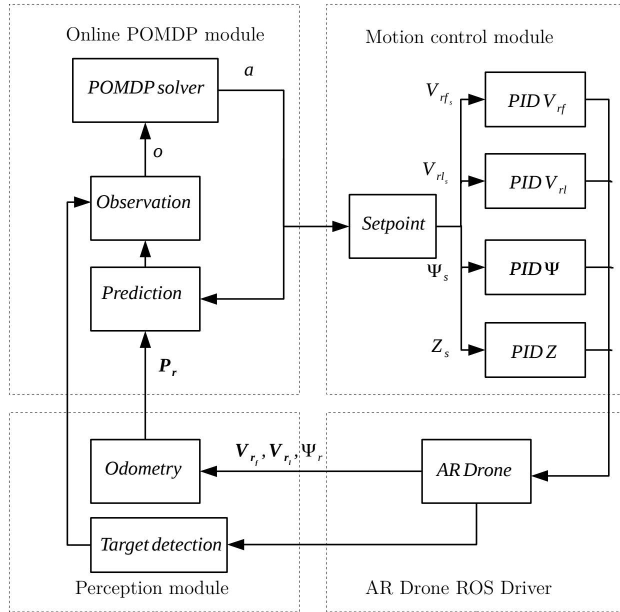


Fig. 1. POMDP ROS System Architecture

Drone Autonomy lab driver, which transforms them into control signals that are sent to the quad-rotor. The motion control module executes in a continuous loop and updates the references for each degree of freedom every time the online POMDP module outputs a new action.

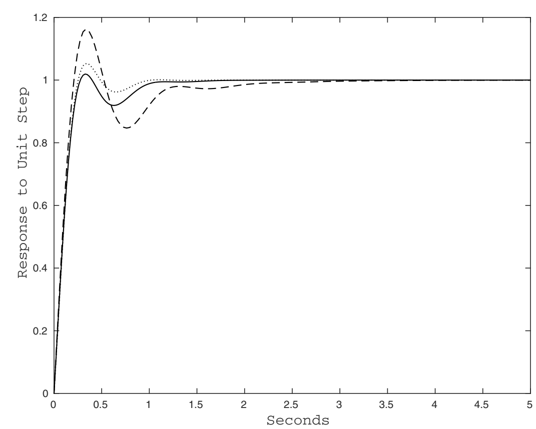


Fig. 2. PID time responses to a unit step input for –$\boldsymbol{V_f}$ and $\boldsymbol{V_l}$, ...Ψ , and - - z. All PID controllers reach steady state within $1s$.

C. *Perception Module*

The perception module is also implemented in ROS and executes at 100 Hz. Current forward and lateral velocity are calculated by fusion of optical flow (using a downward looking camera) and inertial measurement unit readings, yaw angle is calculated based on onboard IMU and altitude is sensed by combining IMU, ultrasonic sensors and barometric sensor.

This module is constantly calculating the current multi-rotor position based on the sensed forward V_{r_f} and lateral V_{r_l} velocities and the heading angle Ψ_r. It converts the forward and lateral velocities in the multi-rotor's frame to the fix world frame, and calculates x_r and y_r positions, using Equation (1). It also reads the altitude or z_r position from the on-board sensors.

In order to have an accurate source of global localisation the system uses Augmented Reality tags that can be detected by the front camera onboard the UAV. This tags are placed on one face of the obstacles and its position in the world frame is known to the system so that when detected they provide an accurate measuring of the UAV pose in the world frame. These tags can only be detected when the UAV is within 1 m in front of the tag. The perception module uses this tags to reset any drift in the odometry once they are detected by the UAV front camera.

$$\begin{bmatrix} x_{r_{t+1}} \\ y_{r_{t+1}} \end{bmatrix} = \begin{bmatrix} x_{r_t} \\ y_{r_t} \end{bmatrix} + \begin{bmatrix} \cos(\Psi_{r_t}) & -\sin(\Psi_{r_t}) \\ \sin(\Psi_{r_t}) & \cos(\Psi_{r_t}) \end{bmatrix} \begin{bmatrix} V_{r_{f_t}} \\ V_{r_{l_t}} \end{bmatrix} \Delta t \tag{1}$$

IV. Target Finding and Tracking

A. Problem Description and Formulation

We illustrate the use of the system for a target finding and tracking scenario in which a multi-rotor UAV flying in a 3D space without access to external GPS localisation and in the presence of obstacles must search and find a ground moving target, and follow it for at least 10 s. Figure 3 shows the flying environment, the target, the obstacles and the UAV. The target and moves on the ground in any direction at an irregular speed. A map of the environment with the obstacles location is known to the system and is input as a parameter of the algorithm as a text file.

After take off, the UAV hovers for a few seconds to initialise its orientation with readings from its on-board sensors. There is some initial drift produced in this initial hovering position and a normal probability distribution with mean value around the take off position is used to model this initial uncertainty.

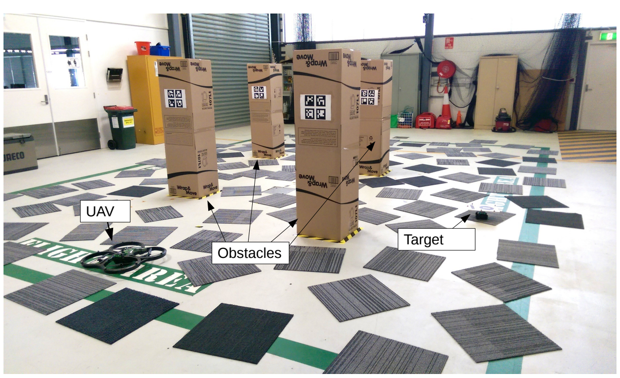


Fig. 3. Target Finding and Tracking scenario

TABLE I
Summary of quad-rotor actions in Target Finding and Tracking mission

Action a	**Forward velocity** V_{r_f} $\frac{m}{s}$	**Lateral velocity** V_{r_l} $\frac{m}{s}$	**Heading change** $\Delta\Psi_a$ $^\circ$	**Altitude change** Δz_r m
Hover	0	0	0	0
Forward	0.6	0	0	0
Backward	-0.6	0	0	0
Up	0	0	0	0.3
Down	0	0	0	-0.3
Roll left	0	0.6	0	0
Roll right	0	-0.6	0	0

The problem is formulated as a POMDP that has the following elements: the state of the aircraft in the environment (S), the set of actions that the multi-rotor can execute (A), the transition function describing the state transition after applying a specific action (T), the observation model that represents the sensed state of the aircraft after taking an action (O), and the reward and cost function (R).

B. State Variables (S)

The state variables considered in the POMDP formulation are the quad-rotor position and heading $\boldsymbol{P_r} = (x_r, y_r, z_r, \Psi_r)$, the target planar position and heading $\boldsymbol{P_t} = (x_t, y_t, \Psi_t)$, target forward velocity $\boldsymbol{V_t}$ and the UAV's velocity $\boldsymbol{V_r}$, all measured in the world frame. The aircraft velocity can be decomposed into two components in the UAV's frame, forward velocity V_{r_f} and lateral velocity V_{r_l}.

C. Actions (A)

The multi-copter actions are designed to take advantage of the four PID controllers, thus the UAV can actuate in four state variables: forward V_{r_f} and lateral V_{r_l} velocities, heading angle Ψ_r and altitude z_r.

The set of actions consists of 7 actions as shown in Fig. I. An action to keep the aircraft hovering, i.e. $\boldsymbol{V_r} = 0\ \frac{m}{s}$; two actions to go forward and backward, with current heading angle Ψ_r, lateral velocity $V_{r_l} = 0\ \frac{m}{s}$, and forward velocity, $V_{r_f} = 0.6\ \frac{m}{s}$ and $V_{r_f} = -0.6\ \frac{m}{s}$, respectively. Actions Up and Down, increase or decrease altitude z_r in $0.3\ m$, respectively, with multi-rotor velocity fixed at $0\ \frac{m}{s}$, and two actions to roll left and right with current heading angle Ψ_r and forward velocity at $V_{r_f} = 0\ \frac{m}{s}$, and lateral velocity at $V_{r_l} = \pm 0.6\ \frac{m}{s}$, respectively.

D. Transition Function (T)

The transition function (T) is based on the set of actions described in Table I. These actions are step inputs or references to the four states controllers. This allows to incorporate step responses, that are acquired experimentally, into the kinematic model of the aircraft using a decoupled model.

The kinematic model is described by Equation (2) and (3). The next multi-rotor position is calculated by obtaining the change in position taking into account the system step responses, the initial and requested values for state variables and the action execution step time. A transformation from the aircraft's frame to the world frame is also calculated in these equations.

The orientation of the aircraft is determined by its heading angle Ψ_r. The uncertainty in motion is included in the system by adding a small deviation to the heading angle σ_r using a normal probability distribution with mean value equal to the desired heading and within the range $-2.0^\circ < \sigma_a < 2.0^\circ$, which represents the uncertainty in the yaw angle when executed by the control system.

$$\begin{bmatrix}\Delta x_{r_t}\\ \Delta y_{r_t}\\ \Delta z_{r_t}\end{bmatrix} = \begin{bmatrix}\cos(\Psi_{r_t}+\sigma_{r_t}) & -\sin(\Psi_{r_t}+\sigma_{r_t}) & 0\\ \sin(\Psi_{r_t}+\sigma_{r_t}) & \cos(\Psi_{r_t}+\sigma_{r_t}) & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}V_{r_{f_t}}\Delta t\\ V_{r_{l_t}}\Delta t\\ \Delta z_r\end{bmatrix} \tag{2}$$

$$\begin{bmatrix}x_{r_{t+1}}\\ y_{r_{t+1}}\\ z_{r_{t+1}}\end{bmatrix} = \begin{bmatrix}x_{r_t}\\ y_{r_t}\\ z_{r_t}\end{bmatrix} + \begin{bmatrix}\Delta x_{r_t}\\ \Delta y_{r_t}\\ \Delta z_{r_t}\end{bmatrix} \tag{3}$$

Where x_{r_t}, y_{r_t} and z_{r_t} are the x, y and z aircraft coordinates at time t, $V_{r_{f_t}}$ and $V_{r_{l_t}}$ are forward and lateral velocities in the quad-rotor's frame at time t, and Ψ_{r_t} and σ_{r_t} are heading and heading deviation at time t.

The transient responses for the four controlled states that are described by Fig. 2 are discretised and included in the transition function as lookup tables in order to approximate the dynamics of the quadrotor.

We used an RC toy car moving on the ground with a marker on top as the target. The target's motion is described by equations 4 and 5. The target's velocity V_{T_t} has an uncertainty of $0.2\frac{m}{s}$ around an average value $\mu = 0.5\frac{m}{s}$. The target behaviour has three modes which are 1)Loop clockwise around the whole flying area which changes the heading of the target Ψ_{T_t}, alternating among north, east, south and west, 2) Same loop as in 1) but around half of the flying area and 3)Continue with same heading with a 50% probability and 50% chance of changing its orientation Ψ_{T_t} to the opposite direction of the UAV in order to go away from it.

$$x_{T_{t+1}} = x_{T_t} + V_{T_t}\cos(\Psi_{T_t})\Delta t \tag{4}$$

$$y_{T_{t+1}} = y_{T_t} + V_{T_t}\sin(\Psi_{T_t})\Delta t \tag{5}$$

E. Observation Model (O)

An observation is composed of the UAV position in the world frame, the target's position if it is detected by the downward looking camera and boolean variables indicating whether the target has been detected and whether or not an AR tag is detected by the front camera. The system receives an odometry reading from the perception module with an uncertainty that is caused by the accumulating error and drift in the yaw angle reading. This uncertainty is represented by adding noise UAV position using a gaussian distribution as in equation (6), with mean value μ around the perceived position and standard deviation $\sigma = 0.5m$.

TABLE II

SUMMARY OF THE REWARD AND COST FUNCTION FOR THE TARGET FINDING AND TRACKING MISSION.

Reward/Cost	**Value**
Detecting the target	500
Hitting an obstacle	-70
Out of region	-70
Movement	-10

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}}e^{-(x-\mu)^2/2\sigma^2} \tag{6}$$

This uncertainty can be reduced, i.e. $\sigma = 0.05m$ if the UAV flies in front of an obstacle where it can detect an AR tag an can reset the error inherent to the odometry and onboard sensors.

If the target is detected by the onboard downward looking camera, the detection system provides the target position within the image. This position is transformed into a position in the world frame. A model of the downward camera field of view is also included in order to extract the target pose in the world frame from the target's position and orientation in the image.

In order for ABT to verify if two observations are equal, a function is included in the code. For two observations to be equal the euclidean distance between the UAV position of both observations should be less than a threshold value that represent the error in the odometry system and the boolean variable that indicates whether a target has been found should have the same value on both observations.

$$Obs_1 = Obs_2 \text{ if } \|P_{r_{Obs_1}} - P_{r_{Obs_2}}\| \leq \epsilon \tag{7}$$

$$\text{and } TO_{Obs_1} = TO_{Obs_2} \tag{8}$$

where $P_{r_{Obs_1}}$ is the quadrotor position in observation 1 and TO_{Obs_1} is the variable that indicates whether a target has been found in observation 1.

F. Rewards and Costs Function (R)

The multi-rotor receives a high reward if it detects the target within the downwards looking camera field of view. Hitting an obstacle or going out of the scenario incur a penalty and every other movement will carry a small cost with the purpose of generating shorter sequences of actions and thus shorter paths. A summary of the reward and cost function is shown in Table II. The values of the reward and cost functions were selected as a result of tuning the system to be able to accomplish the mission faster.

V. RESULTS AND DISCUSSION

We conducted simulation and real flight test in order to analyse the performance of the system.

TABLE III

Simulation results for Target finding problem

Target's mode (x, y, z)	**Success rate (%)**	**Flight time to target** (s) **(Number of steps)**	**Discounted return**
Small loop	100	27	1840
Big loop	100	43	1023
Target escaping	100	93	103

A. Simulation

The target finding and tracking with UAV mission was tested in simulation for 100 runs for each of the three cases shown in table III to get an average for the discounted return, the number of steps and the success to accomplish the mission with a maximum of 120 steps which corresponds to 2 min flight time.

Simulation results indicate that the system is able to find a target and follow it in all three cases. If the target behaviour is mostly deterministic as in the two first cases, small loop and big loop the return is higher and the number of steps to accomplish the mission is lower. On the other hand, when the uncertainty in the target's behaviour increases, the system takes more steps to accomplish the mission.

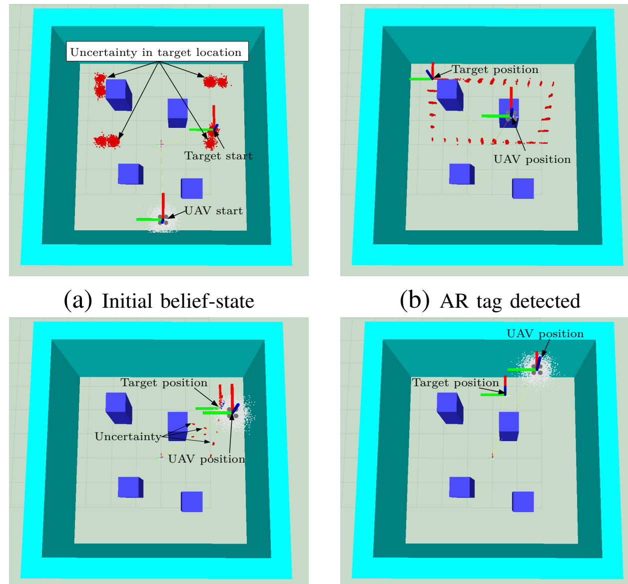
We use the graphical tool RVIZ for visualisation (Fig. 4). Point clouds are used to visualise the distribution of the belief-state particles representing the target and UAV locations. The target starts moving from four possible locations. The UAV is initially located at coordinates $(-2.9, 0, 0.0)$ and takes off and hovers for some seconds which increases the uncertainty in its position, seen as white particles in Fig. 4(a). In Fig. 4(b) the UAV detects an AR tag on an obstacle with its front camera and as a result its position uncertainty is reduced (see white particles). In Fig. Fig. 4(d) the UAV detects the target with its downward looking camera, which reduces the uncertainty in the target position seen as red particles.

In the target escaping mode, the system and POMDP solver guide the multi-rotor towards the corners of the flying space and spends some time exploring these regions, see Fig. 4(c). After exploring a corner the algorithm moves the quad-rotor away from the boundaries of the space towards the centre and then to the other corners, trying to avoid exiting the flying space.

B. Real Flight Tests

Real flight test were conducted to explore the performance of the system under real conditions. The experiments were conducted using a low cost commercial platform, the Parrot AR Drone, with four obstacles, Fig. 3.

Real flight tests were conducted 20 times for each of the target modes. Table IV shows that the system successfully completes the mission 90% of the times for the small loop and 80% for the bigger loop. On the other hand, the system has more difficulty finding a mobile target that is trying to escape and completes the mission 55% of the times.



(a) Initial belief-state (b) AR tag detected

(c) Target followed (d) Target detected

Fig. 4. Example of UAV trajectories (orange) for four different Target (black) starting locations.

TABLE IV

Real flight results for Target finding problem

Target's mode (x, y, z)	**Success rate (%)**	**Flight time to target** (s) **(Number of steps)**	**Discounted return**
Small loop	90	48	893
Big loop	85	57	486
Target escaping	55	107	54

The difference in success rate compared to the simulation results is due to the nature of the target motion, which in the real scenario gets affected by the tiles that are placed on the ground to generate texture. This creates a larger stochastic component on the target motion that is difficult to model. Furthermore, the target is a toy car that is driven remotely by a person which also contributes to generating an unpredictable component to its motion.

An example of the paths followed by the UAV and target is shown in Fig. 5, in this case the UAV flies first near an obstacle to reduce the uncertainty in its position and then it ascends to have a wider FOV and detects the target. Afterwards, the UAV follows the target until it is detected for 10 times and finishes the mission.

The system is also able to re-capture the target if the target goes out of the field of view of the camera. The POMDP formulation makes the system robust to predict the target's next possible location to navigate towards it to detect it. This situation is shown in Fig. 5. It can be seen that the UAV was able to find, detect and track the target in a real flight scenario.

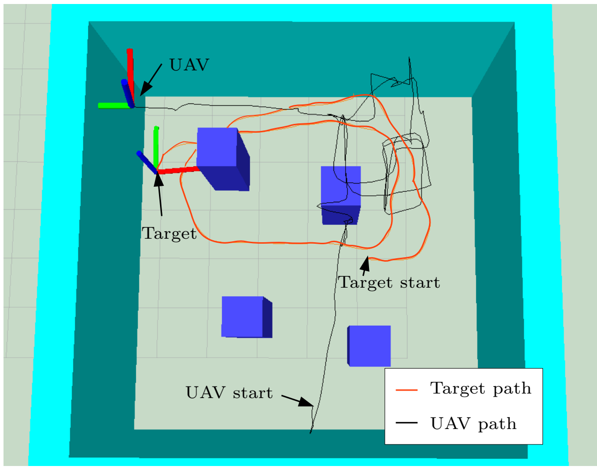


Fig. 5. Real flight UAV trajectory (black) and Target trajectory (orange)

VI. Conclusions

A system for target finding and tracking with UAVs using POMDP in GPS-Denied and cluttered environments was presented and demonstrated in simulation and flight test, this system is able to find, detect and track a target with a UAV flying within a confined space using an onboard downwards looking camera and in the presence of obstacles.

The system presented uses a state-of-the-art POMDP online algorithm that outputs actions instead of waypoints, and only relies on on-board sensors for localisation of both, the UAV and the Target. This allows modelling the system's dynamics using the motion controller responses for each decoupled state, in this case forward and lateral velocities, altitude and yaw angle.

The system is also able to find, to track and to follow a target in a real scenario when the target's motion model is mostly deterministic. The system is capable of following a target when it is pursued by an UAV in some cases, even if the target is lost the UAV can search and reacquire the targets position in some cases.

Current ongoing work focuses on outdoor testing and on exploration of the response of the system in an environment with windy conditions and on the exploration of different reward functions.

ACKNOWLEDGMENT

The authors thank ARCAA for the use of its hangar facilities for indoor flight tests. We thank Hanna Kurniawati for her fruitful discussion on POMDP as well as her support and for the open source software ABT.

References

[1] C. P. C. Chanel, F. Teichteil-Knigsbuch, and C. Lesire, Multi-Target Detection and Recognition by UAVs Using Online POMDPs., in AAAI, 2013.

[2] C. P. C. Chanel, F. Teichteil-Knigsbuch, and C. Lesire, Multi-Target Detection and Recognition by UAVs Using Online POMDPs., in AAAI, 2013.

[3] R. He, A. Bachrach, and N. Roy, Efficient planning under uncertainty for a target-tracking micro-aerial vehicle, in Robotics and Automation (ICRA), 2010 IEEE International Conference on, 2010, pp. 18.

[4] U. Zengin and A. Dogan, Real-Time Target Tracking for Autonomous UAVs in Adversarial Environments: A Gradient Search Algorithm, IEEE Transactions on Robotics, vol. 23, no. 2, pp. 294307, Apr. 2007.

[5] S. A. P. Quintero and J. P. Hespanha, Vision-based target tracking with a small UAV: Optimization-based control strategies, Control Engineering Practice, vol. 32, pp. 2842, Nov. 2014.

[6] A. Rucco, A. P. Aguiar, and J. Hauser, Trajectory Optimization for Constrained UAVs: a Virtual Target Vehicle Approach, in Unmanned Aircraft Systems (ICUAS), 2015 International Conference on, 2015, pp. 236245.

[7] Eich, Markus, Francisco BonninPascual, Emilio GarciaFidalgo, Alberto Ortiz, Gabriele Bruzzone, Yannis Koveos, and Frank Kirchner. "A robot application for marine vessel inspection." Journal of Field Robotics 31, no. 2 (2014): 319-341.

[8] K. Krishnamoorthy, D. Casbeer, and M. Pachter, Minimum time UAV pursuit of a moving ground target using partial information, in Unmanned Aircraft Systems (ICUAS), 2015 International Conference on, 2015, pp. 204208.

[9] AR Drone Autonomy Lab. A Ros Driver for the Parrot AR Drone UAV. http://ardrone-autonomy.readthedocs.org/en/latest/, 2014.

[10] D. Hsu, W. S. Lee, and N. Rong, A point-based POMDP planner for target tracking, in Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on, 2008, pp. 2644-2650.

[11] H. Kurniawati, D. Hsu, and W. S. Lee, SARSOP: Efficient Point-Based POMDP Planning by Approximating Optimally Reachable belief-state Spaces, Proceedings of Robotics: Science & Systems, 2008.

[12] J. Ballesteros, L. Merino, M. A. Trujillo, A. Viguria, and A. Ollero, Improving the efficiency of online POMDPs by using belief-state similarity measures, in Robotics and Automation (ICRA), 2013 IEEE International Conference on, 2013, pp. 1792-1798.

[13] J. Capitan, M. T. J. Spaan, L. Merino, and A. Ollero, Decentralized multi-robot cooperation with auctioned POMDPs, The International Journal of Robotics Research, vol. 32, no. 6, pp. 650-671, May 2013.

[14] H. Kurniawati and V. Yadav, An Online POMDP Solver for Uncertainty Planning in Dynamic Environment, in International Symposium on Robotics Research, 2013.

[15] N. Dadkhah and B. Mettler, Survey of Motion Planning Literature in the Presence of Uncertainty: Considerations for UAV Guidance, Journal of Intelligent & Robotic Systems, vol. 65, no. 14, pp. 233246, Jan. 2012.

[16] H. Kurniawati, Y. Du, D. Hsu, and W. S. Lee, "Motion planning under uncertainty for robotic tasks with long time horizons," The International Journal of Robotics Research, vol. 30, pp. 308-323, 2011.

[17] S. C. W. Ong, S. W. Png, D. Hsu, and W. S. Lee, Planning under uncertainty for robotic tasks with mixed observability, The International Journal of Robotics Research, vol. 29, no. 8, pp. 10531068, 2010.

[18] Papadimitriou, C.H., and J.N. Tsitsiklis. The Complexity of Markov Decision Processes. Mathematics of Operations Research 12, no. 3 (1987): 44150.

[19] Thrun, S., W. Burgard, and D. Fox. Probabilistic Robotics. Vol. 1. MIT press Cambridge, MA, 2005.

[20] J. Pineau, G. Gordon, and S. Thrun, Anytime point-based approximations for large POMDPs, Journal of Artificial Intelligence Research, vol. 27, no. 1, pp. 335380, 2006.

[21] D. Silver and J. Veness, Monte-Carlo Planning in large POMDPs, Advances in Neural Information Processing Systems (NIPS), vol. 46, 2010.

[22] D. Klimenko, J. Song, and H. Kurniawati, TAPIR: A Software Toolkit for Approximating and Adapting POMDP Solutions Online.

[23] W. H. Al-Sabban, L. F. Gonzalez, and R. N. Smith, Extending persistent monitoring by combining ocean models and Markov decision processes, presented at the Proceedings of the 2012 MTS/IEEE Oceans Conference, 2012.

[24] W. H. Al-Sabban, L. F. Gonzalez, and R. N. Smith, Wind-Energy based Path Planning For Unmanned Aerial Vehicles Using Markov Decision Processes, presented at the ICRA 2013, 2013.