Open Archive Toulouse Archive Ouverte (OATAO)

OATAO is an open access repository that collects the work of some Toulouse researchers and makes it freely available over the web where possible.

This is an author’s version published in: https://oatao.univ-toulouse.fr/19444


To cite this version:

Delamer, Jane-Alexis and Watanabe, Yoko and Ponzoni Carvalho Chanel, Caroline MOMDP modeling for UAV safe path planning in an urban environment. (2017) In: Journées Francophones sur la Planification, la Décision et l'Apprentissage pour la conduite de systèmes (JFPDA 2017), 6 July 2017 - 7 July 2017 (Caen, France).

Any correspondence concerning this service should be sent to the repository administrator:
tech-oatao@listes-diff.inp-toulouse.fr
MOMDP modeling for UAV safe path planning in an urban environment

Jean-Alexis Delamer\textsuperscript{1,2}, Yoko Watanabe\textsuperscript{1}, and Caroline P.Carvalho Chanel\textsuperscript{2}

\textsuperscript{1}ONERA - The French Aerospace Lab, Information Processing and Systems Departement (DTIS)
\textsuperscript{2}Université de Toulouse, ISAE-SUPAERO Institut Supérieur de l’Aéronautique et de l’Espace, Design and Control of Aerospace Vehicles Departement (DCAS)

Abstract

Path planning is a research domain very active and applied among others on autonomous vehicle such as UAV. In recent years, a lot of progress has been made on path planning under uncertainties issued by a vehicle navigation system, for example in localization or environment mapping. However, such uncertainties are often treated by the path planner in a deterministic way. That is, the navigation system’s performance is deterministically given in function of the environment. This paper tackles a more complex problem of UAV safe path planning in an urban environment, in which UAV is at risks of GPS signal occlusion and obstacle collision. The key idea is to make a UAV path planning along with its navigation and guidance mode planning, where each of such mode uses different set of sensors and whose availability and performance are environment-dependent. A partial knowledge on the environment is supposed to be available, in a form of probability maps of obstacles and sensor availabilities. To solve this problem the UAV need to be well represented in the planner model and so do the associated uncertainty propagation. This paper proposes a model based on Mixed Observability Markov Decision Process (MOMDP). The proposed MOMDP model allows the planner to choose the best path-direction with the adapted sensor set for an UAV to reach a mission goal efficiently and safely. This paper only provides a MOMDP model for the planner, and the planning algorithm and preliminary results will be expected in the final paper.

1 Introduction

These last years there has been a growing need for UAVs (Unmanned Aerial Vehicles) to accomplish distant mission in cluttered environments (urban areas with high presence of building, ...). Such missions can be done only if the safety conditions are gather, and especially the planned path must be collision risk free. The safety of the UAV depends on its onboard navigation capabilities (including onboard sensor performances) as well as on the environment. Moreover the sensors availability and performance depends on the environment, for example GPS is particularly dependent of the quality of the signal (occlusion / degradation) and especially in cluttered environment. All this parameter affect the navigation performance and must be taking into account in path planning to ensure the success of the mission.

Therefore, the community have proposed different frameworks and algorithms to solve path planning under uncertainty in cluttered and continuous environment. A method was proposed by [1] to compute an efficient collision-free path for MAV (Micro Aerial Vehicles) while taking into account the dynamic of the MAV. This vehicle has onboard camera to estimate its localization and the path is computed based on the algorithm RRBT (Rapidly-exploring Random Belief Trees [6]) which allows to consider position uncertainty during planning. The autonomous vehicles can have several onboard sensors to ensure the safety of the vehicles and its passengers. Another path planning approach has been proposed for autonomous vehicles in unknown semi-structured environment [7] by using an hybride-state A* search algorithm. Note that these previous frameworks compute a static path (a plan and not a conditional plan or policy). When considering a dynamic path which is able to self-adapt in function of the events, some frameworks were also proposed by the community. One could cite, the work of [5] that propose a method based on POMDP (Partially observable Markov decision process)
for autonomous vehicles in real situations. Thus this approach take into account other vehicles to take its decisions (changing lane, break, ...).

All these works are based on various sensors to enable the vehicle to guide itself and compute the best path. One limitation of these works, in the case of UAV (or MAV) is that they do not consider path planning in urban environment. The number of onboard sensors are often relatively low and it do not take into account the availability of the sensors as for example, their precision in function of the environment and there is only one navigation mode (using all the sensors).

In this paper, it is proposed a path planner model that will ensure an efficient risk-free path. To success, it will be take into account the dynamic of the UAV, and the availability of sensors in the environment through a set of availability map. The sensor availability maps are probability grids which are in overlay of the environment map, which is not discretized (Figure 3b). Like the sensor availability maps, the obstacles map is a grid in overlay of the environment map. The path computed called a policy will allow at each instant the UAV to take the best action in function of the sensor’s availability. Depending on the availability of the sensors, different navigation modes could be used, resulting in different localization and execution error propagation. This point should be considered in path computation. So, in this problem the UAV have an inertial navigation system (INS), a GPS, an optical flow field measurement and a landmark pixel coordinate measurement to navigate. The UAV have two guidance modes, the waypoint tracking whose precision depends on the navigation mode used and the wall following mode. These sensors and guidance modes allow the planner to optimize the combination sensors/guidance mode for each event that can occur.

Navigate through an cluttered environment can be hard for an UAV due to the proximity of the obstacle and the great variance on the availability of the sensors. Therefore the objective of this works are to find a plan (or policy in MOMDP) that take into account the availability of the onboard sensors to ensure the shortest collision-free path. The policy must take in entry the belief on the actual state and return the best action to execute. The main idea is to combine the transition function of the decision process to the error (localization and execution) propagation function, which depends on the localisation and guidance mode for each part of the path. The advantage of this method is to incorporate a priori knowledges on the disponibility of the sensor in the path planning, and to propagate this informations on the futur path. This allows the algorithm to calculate a path to guide the vehicle to the safest and shortest path. Taking account of the previous idea, and with our uncertainty model, we have chosen to lean our model on the Markov Decision Process [3]. Given the nature of our problem and the partially informations on the state of the system, we will use the MDP extension: Mixed Observability Markov Decision Processes (MOMDP) [10].

The paper is organized as follows: in Section 2 it will be explained the problem including the system architecture, the GNC’s transition model and details on the maps used. In Section 3 it will be recalled the definition of the MOMDP and then it will be defined the MOMDP planning model. Finally, in Section 4 it will be given the perspectives of the future works and a conclusion.

2 Problem statement

2.1 System architecture and time differentiation

The architecture of the overall system considered in this problem combine the GNC’s transition model which includes the vehicle motion model, onboard sensor model and flight control systems, and the MOMDP planning model. The planner will consider GNC’s transitions, which are known to compute the possible evolutions of the system’s state. The policy given by the MOMDP planner take as inputs the probability distribution over the actual state \( b_0 \), and a vector of boolean on sensors availability \( s'_v \).

And it will return an action, which will select the direction and the navigation and guidance modes to perform during a certain amount of time, a new vector of sensor’s availability is observed, and a belief state update is performed. Again this new belief state is used by the policy to define the next action. During the following article, the diagram presented in Figure 1 will be referred to facilitate understanding.

The formalisms such as the MDPs, in their majority and most of the variants, consider the actions to accomplish instantly and not as a process during over the time. There is a variant, the semi-Markov Decision Problems (SMDP), which apply the MDP to the continuous time, where each actions have

\(^1\)Guidance, Navigation and Control (GNC)
a execution time [4]. However, in this case, the problem is not that actions are durative, but the state of the UAV changes during an action. Our action include a navigation mode used to locate the UAV during the entire movement in the direction \( d \).

In this sense, it will be distinguished two unit of time due to the difference between the unit of time of the GNC’s transition model and the time unit of the MOMDP planning model named \( epoch \). The MOMDP planning model works at a lower frequency that the system, thus an epoch is equivalent to several unit of time of the system’s transition model. The reason is to lower the complexity of the algorithm by reducing the total number of actions to complete the task.

To differentiate the unit of time, the GNC unit of time will be noted \( k \) and the epoch of the MOMDP will be noted \( t \) (Figure 2).

\[
\Phi^k = \begin{bmatrix} 1 & \Delta t & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad B^k = \begin{bmatrix} 0 \\ \frac{\Delta t^2}{2} I \\ 0 \end{bmatrix}, \quad Q^k = \Delta t \tilde{Q}
\]

Figure 1: Architecture diagram

Figure 2: Difference between the units of time

2.2 GNC’s transition model

As stated earlier and shown in Figure 1, the system’s transition model consists of the vehicle motion model, onboard sensor model, and the UAV flight guidance, navigation and control (GNC) system. This section presents these models which construct a state transition model used in by the planning model.

2.2.1 State transition model

The state \( x \) of the UAV is defined by its positon, its velocity and the accelerometer bias such as \( x = [X^T \ V^T \ b_a]^T \), where \( X, V \) and \( b_a \) are respectively the UAV position, the velocity and the accelerometer bias. Then the state transition can be defined such as :

\[
\dot{x} = \begin{bmatrix} \dot{X} \\ \dot{V} \\ \dot{b_a} \end{bmatrix} = \begin{bmatrix} a \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} v_x \\ v_v \\ v_{ba} \end{bmatrix} = \begin{bmatrix} 0 & I & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ I \\ 0 \end{bmatrix} a + v = Ax + Ba + v \quad (1)
\]

where \( a \) is the acceleration and \( v \sim N(0, \tilde{Q}) \) is the process noise. According to this model, the state transition from \( x(t_k) = x_k \) to \( x(t_{k+1} = t_k + \Delta t) = x_{k+1} \) can be derived as:

\[
x_{k+1} = \Phi x_k + Ba_k + v_{k+1} \quad (2)
\]

where \( v_{k+1} \sim N(0, Q) \) is the discretized process noise and

\[
\Phi = \begin{bmatrix} I & \Delta t I & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix}, \quad B = \begin{bmatrix} \frac{\Delta t^2 I}{2} \\ \Delta t I \\ 0 \end{bmatrix}, \quad Q = \Delta t \tilde{Q}
\]
2.2.2 State estimator

A true value of the vehicle state $x$ is never accessible in reality, and so is estimated by the navigation module by using sensor measurements available onboard (Figure 1). This problem supposes the state estimator based on an EKF (Extended Kalman Filter [15]) which includes two steps:

i. the INS prediction

ii. the sensor measure (if available) used for correction

The INS measurement integration enables a high frequency state estimation, but it suffers from the estimation drift due to the measurement bias. In order to limit or even correct such drift, other sensors are fused with INS through the second step of Kalman filter. The most common and effective sensor is GPS (or other similar satellite-based navigation system) which can provide an accurate absolute position measurement of an UAV. But it is at a high risk of its performance degradation due to multi-path effect or signal occlusion in a cluttered environment. Alternative approaches proposed by the robotics community (particularly for indoor UAVs) are based on onboard vision information. Such approaches include visual odometry and SLAM algorithms for motion estimation, or image matching with geo-referenced map (e.g. image database, landmarks) for absolute localization. As an example, we consider GPS and vision-based landmark detection as navigation sensors in this paper.

**INS Prediction**: The accelerometer measurement $a_{IMU}$ is used to propagate the estimated state. It measures the biased, non-gravitational UAV body acceleration

$$a_{IMU} = R_{BIk} (a_k - g) + b_k + \xi_{IMU}$$

where $R_{BIk}$ is a rotation matrix from the inertial to the UAV body frames (assumed to be known), $g$ is the gravity vector and $\xi_{IMU} \sim N(0, R_{IMU})$ is the IMU measurement error. According to the process model (Eq 2), the estimated state $\hat{x}_k$ is propagated to

$$\hat{x}_{k+1} = \Phi \hat{x}_k + B \left( R_{BIk}^T B_{IMU} - \hat{b}_k \right) + g$$

$$= \Phi \hat{x}_k + B \left( a_k + R_{BIk}^T B_{IMU} \right) + \xi_{IMU}$$

$$= x_{k+1} - \Phi \hat{x}_k - v_{k+1} + BR_{BIk}^T b_k + \xi_{IMU}$$

Then the state prediction error is given by:

$$\tilde{x}_{k+1} = x_{k+1} - \hat{x}_{k+1} = \Phi \tilde{x}_k + v_{k+1} - BR_{BIk}^T b_k + \xi_{IMU}$$

$$= (\Phi - \Delta \Phi_k^e) \tilde{x}_k + v_{k+1} - BR_{BIk}^T \xi_{IMU}$$

where $\Delta \Phi_k^e = BR_{BIk}^T \left[ 0 \ 0 \ I \right]$. The associated error covariance is given by:

$$P_{k+1} = (\Phi - \Delta \Phi_k^e) P_k (\Phi - \Delta \Phi_k^e)^T + Q + \tilde{R}_{IMU}$$

where $\tilde{R}_{IMU} = BR_{BIk}^T R_{IMU} R_{BIk} B^T$. For simplicity, we can consider the case of $R_{IMU} = \sigma_{IMU}^2 I$ and hence $\tilde{R}_{IMU} = BR_{BIk} R_{BIk} B^T$ remains constant for all $k$.

**GPS correction**: When GPS is available at $t_{k+1}$, the predicted state (Eq: 5) can be corrected by using its position and velocity measurement $z_{GPS_{k+1}}$.

$$z_{GPS_{k+1}} = \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \end{bmatrix} x_{k+1} + \xi_{GPS_{k+1}} = H_{GPS} \tilde{x}_{k+1} + \xi_{GPS_{k+1}}$$

where $\xi_{GPS} \sim N(0, R_{GPS})$ is the GPS measurement error. Then, the GPS measurement is used to correct the estimated state such as:

$$\hat{x}_{k+1} = \hat{x}_{k+1} + K_{GPS_{k+1}} H_{GPS} \left( z_{GPS_{k+1}} - H_{GPS} \hat{x}_{k+1} \right)$$

$$= \hat{x}_{k+1} + K_{GPS_{k+1}} H_{GPS} \tilde{x}_{k+1} + K_{GPS_{k+1}} \xi_{GPS_{k+1}}$$

$$= x_{k+1} - (I - K_{GPS_{k+1}} H_{GPS}) \tilde{x}_{k+1} + K_{GPS_{k+1}} \xi_{GPS_{k+1}}$$

$$= x_{k+1} - \left((I - K_{GPS_{k+1}} H_{GPS}) \Phi \hat{x}_k + v_{k+1} - BR_{BIk}^T b_k + \xi_{IMU} \right)$$

$$= x_{k+1} - \Phi \hat{x}_k - v_{k+1} + BR_{BIk}^T b_k + \xi_{IMU}$$

$$= \hat{x}_{k+1}$$
where $K_{GPS_{k+1}} = P_{k+1}^{-1} H_{GPS}^T (H_{GPS} P_{k+1}^{-1} H_{GPS}^T + R_{GPS})^{-1}$ is a Kalman gain. Then the error estimate and its covariance are given by:

$$
\hat{x}_{k+1} = x_{k+1} - \hat{x}_{k+1} = (I - K_{GPS_{k+1}} H_{GPS}) \hat{x}_{k+1} - K_{GPS_{k+1}} \xi_{GPS_{k+1}}
$$

$$
P_{k+1} = (I - K_{GPS_{k+1}} H_{GPS}) P_{k+1}^{-}
$$

(8)

**Landmark correction:** When a landmark (whose position $x_{LM}$ is a-priori given) is visible and detectable on an onboard camera image at $t_{k+1}$, its pixel-coordinates information can be used to correct the predicted state (Eq: 5). By assuming a pin-hole camera model, the pixel-coordinates measurement is given by the following nonlinear measurement model.

$$
z_{LM_{k+1}} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \end{bmatrix} \frac{X_{LM_{k+1}}}{e_3} + \xi_{LM_{k+1}} = C x_{LM_{k+1}} + \xi_{LM_{k+1}} = h_{LM}(x_{k+1}) + \xi_{LM_{k+1}}
$$

(9)

where $C$ is a known camera matrix, $X_{LM_{k+1}} = R_{CB} R_{BL_{k+1}} (X_{LM} - X_{k+1})$ and $\xi_{LM} \sim N(0, R_{LM})$ is the landmark image-detection error in pixels. The camera is assumed to be mounted at the c.g. of the UAV with a known camera orientation $R_{CB}$ with respect to the UAV body. An extended Kalman filter can be applied, and similarly to (Eq: 8), the resulting estimation error and its covariance matrix are given by:

$$
\hat{x}_{k+1} = (I - H_{LM_{k+1}} K_{LM_{k+1}}) \hat{x}_{k+1} - H_{LM_{k+1}} K_{LM_{k+1}} \xi_{LM_{k+1}}
$$

$$
P_{k+1} = (I - H_{LM_{k+1}} K_{LM_{k+1}}) P_{k+1}^{-}
$$

(10)

where $H_{LM_{k+1}}$ is a Jacobian matrix of the nonlinear measurement function $h_{LM}(x_{k+1})$ evaluated at $x_{k+1} = \hat{x}_{k+1}$. It should be noted that $H_{LM_{k+1}}$ thus depends on the predicted state $\hat{x}_{k+1}$, while $H_{GPS}$ does not.

**INS-only solution:** If no correction is made with neither GPS nor Landmark detection, the state estimate at $t_{k+1}$ is given by the predicted one:

$$
\hat{x}_{k+1} = \hat{x}_{k+1}^{-}
$$

$$
P_{k+1} = P_{k+1}^{-}
$$

(11)

2.2.3 Guidance laws

A set of actions will be defined in the planner model, such as each action is defined among others as a desired direction of motion control. For example, as the exact definition will be given in subsection 3.3.2, a set of four actions can be defined as: {"Move-to-North", "Move-to-East", "Move-to-South", "Move-to-West"}.

Given a direction and a reference speed $V_{ref}$ is desired velocity in the desired direction the following linear guidance law can be applied to realize the action:

$$
a_k = K_p V_{ref} - K_d \dot{V}_k
$$

(12)

where $K_p, K_d > 0$ are the control gain and $\dot{V}_k$ is the estimated UAV velocity at instant $t_k$.

This paper considers two different guidance modes in terms of information sources used in the guidance law (Eq 12) for $V_k$. The first guidance mode uses the state estimation result from the navigation module given in Section 2.2.2, while the second mode uses some sensor measurements directly. The former mode corresponds to a conventional absolute guidance approach such as WP tracking (Waypoint Tracking), and the latter to a sensor-based relative guidance approach such as visual servoing to follow a wall. The advantage of the sensor-based relative guidance mode is that it is independent from the navigation (i.e., localization) performance, but in return its applicability is rather limited to a proximity of some known object (e.g. wall).
Absolute guidance: In the absolute guidance mode, we have \(\hat{x}_k = [0 \ I \ 0] \hat{x}_k\) where \(\hat{x}_k\) is the estimated state from Section 2.2.2. Then, \(x_{k+1}\) can be obtained by substituting this guidance law (Eq 12) into the discretized process model (Eq 2):

\[
x_{k+1} = \Phi x_k + B(K_p V_{ref} - K_d [0 \ I \ 0] \hat{x}_k) + v_{k+1}
\]

where \(\Delta \Phi = BK_d [0 \ I \ 0]\). Hence, the state \(x_{k+1}\) follows the normal distribution as below.

\[
x_{k+1} \sim N((\Phi - \Delta \Phi x_k + BK_p V_{ref} \Delta \Phi P_k x_k + v_{k+1})
\]

where the covariance \(\tilde{Q}_{k+1}\) is a function of the estimation error covariance \(P_k\).

Sensor-based relative guidance: In the sensor-based relative guidance mode, \(\hat{V}_k\) in (Eq 12) is directly given from some onboard sensors. For example, optical flow information from a video sequence can be used in combination with a distance measurement. Let us assume the measurement error \(\hat{V}_k = (V_k - \hat{V}_k) \sim N(0, R_{V_k})\). Then, similarly to (Eq 13),

\[
x_{k+1} = (\Phi - \Delta \Phi x_k + BK_p V_{ref} + BK_d \hat{V}_k + v_{k+1}
\]

\[
\sim N(\bar{x}_{k+1|k}, BK_d R_{V_k} K_d^T + Q) = N(\bar{x}_{k+1|k}, \tilde{Q}_{k+1})
\]

where the covariance \(\tilde{Q}_{k+1}\) is independent from \(P_k\).

2.2.4 State probability density function

Given the initial state \(x(t_0) = x_0\) and the initial estimation error covariance \(P_0\) which gives \(\bar{x}_0 \sim N(0, P_0)\). As defined later in Section 3.3.2, an action \(a\) in the planner model corresponds to a combination of the direction of desired motion (\(V_{ref}\)), the navigation mode (GPS, Landmark or INS-only) and the guidance mode (Absolute or Sensor-based). For a given action \(a\), it is possible to obtain the distribution of the next state \(x_1 = x(t_0 + \Delta t)\) knowing \(x_0\).

\[
f_x(x_1|x_0) \sim N(\hat{x}_{1|0}, \tilde{Q}_1)
\]

where \(\tilde{Q}_1\) is either \(\tilde{Q}_1\) in (Eq 14) or \(\tilde{Q}_1\) in (Eq 15) depending on the selected guidance mode. At the same time, the state estimation error covariance is updated to \(P_1\) by using the selected navigation mode (Eqs 8, 10 or 11).

Now recall from Section 2.1 that the system’s transition model and the planning model do not have the same time unit. Usually, the planning time step is much longer than that of the system’s transition model. It means that a single state transition \(s = s_0 \rightarrow s_1\) with a selected action \(a\) in the planner corresponds to several consecutive state transitions \(x_0 \rightarrow x_n\) in the system’s transition model. So we have to continue the state transition further up to \(n > 1\) from Eq. 16 and \(P_1\) with the same action \(a\). The conditional state distribution at \(t_k\) \((k > 1)\) knowing \(x_0\) can be obtained sequentially as follows.

\[
f_x(x_k|x_0) = \int f_x(x_k|x_{k-1}) f_x(x_{k-1}|x_0) dx_{k-1}
\]

where \(f_x(x_k|x_{k-1}) \sim N(\hat{x}_{k|k-1}, \tilde{Q}_k)\). In parallel, the Kalman filtering process (of the selected navigation mode) is repeated \(k\) times to obtain \(P_k\). When \(\tilde{Q}_k\) and \(P_k\) do not depend on the state \(x_{k-1}\), the integration above will result in a normal distribution:

\[
f_x(x_k|x_0) \sim N((\Phi - \Delta \Phi x_k + BK_p V_{ref}, (\Phi - \Delta \Phi x_{k-1}) \tilde{Q}_{k-1} x_k + v_{k+1})
\]

\[
\tilde{Q}_k
\]

where \(\bar{x}_{1|0} = (\Phi - \Delta \Phi x_k + BK_p V_{ref}\text{ and } \tilde{Q}_1 = \tilde{Q}_1)\). \(\tilde{Q}_k\) is obtained by (Eq 14 or 15).

The derivation of the state distribution (Eq 17) becomes more complex in a case of having a dependency of \(\tilde{Q}_k\) and \(P_k\) on the state \(x_{k-1}\). In order to avoid this complication, the matrices \(\tilde{Q}_k\) and
\( P_k \) can be approximated by those evaluated at the expected state \( \bar{x}_{k-1|0} \). Then, the state transition function required in the planning model can be given by Eq. 18 when \( k = n \). The decision of the couple direction / navigation mode need that the navigation mode is available. The navigation and guidance modes are necessary during the entire action \( a \), however during the verification of the availability of a mode it is very difficult to know if the sensor will be available for the entire action. It is necessary to precise that in the case where a sensor will not be available during the entire movement, the navigation mode will switch to INS - Only (the only navigation mode available permanently). To simplify our decisional problem, we will suppose that if the sensor is available at the end of the action, the sensor was available during the entire movement. This state transition function from the state \( s = s_0 \) to \( s' = s_1 \) can be re-written with a notation for the planning model as below.

\[
f_S(s' = s_1|s = s_0) = f_X(x_n|x_0) \sim N(\bar{x}_{n|0}, \Sigma_n) = N(s', \Sigma') \tag{18}
\]

It is worth emphasizing here that this equation will be the only link to the GNC’s transition model with the MOMDP planning model (Section 3.3).

### 2.3 Environment Maps

The planner will use a-priori knowledge on the environment in which UAV will navigate. This knowledge is provided as a set of maps including information on obstacles as well as on sensor availabilities(Fig 3b). It is supposed that a 3D environment model of the UAV mission operation site is available beforehand, either from some database, from data obtained from the past missions, or from data acquired in a pre-mission flight at high and safe altitude. This 3D environment model can be represented as a discretized 3D occupancy map, where the 3D space is divided into cells. Let us denote the \( i \)-th cell of the map by \( c_i \). Then a probability that the cell \( c_i \) is occupied, i.e., obstacle, is given as \( p(Collision|c_i) \).

Sensor availability maps are also given in the same form, as a set of probabilities that a sensor is available at each cell \( c_i \). These maps can be generated by considering the corresponding sensor characteristics in relation with the environment (given by the occupancy map). For example, GPS performance suffers from its signal occlusion or multi-path effect due to surrounding obstacles. It is common to measure the performance of GPS by metrics called DOPs (Dilutions of Precision) which corresponds to a standard deviation of the measured position error. Such DOPs can be predicted by a GPS simulator for a specified geo-location, date and time, and environment (obstacles). Figure 3a shows examples of GPS-PDOP (Position Dilution of Precision) map obtained by using OKTAL SE-NAV simulator available at ONERA/DEMR (Dept. of Electro Magnetism and Radar). In this paper, this PDOP map is transformed to GPS availability map by setting a maximum allowable position error threshold \( \epsilon \). In short, GPS is considered to be available if its position error \( \tilde{X}_{GPS} \) is inferior to this threshold. From the corresponding PDOP value, a probability of GPS availability at each cell \( c_i \) is calculated by

\[
p(GPS|c_i) = p(\tilde{X}_{GPS}(c_i) < \epsilon), \quad \tilde{X}_{GPS}(c_i) \sim N(0, PDOP(c_i)).
\]

Availability of the navigation mode using landmark detection at each cell is obtained in function of the camera’s field-of-view and detection performance of the image processor. It is straightforward from the pin-hole camera model (Eq 9) to judge whether a given landmark position lies within the camera’s field-of-view from a given cell position. The image-detection rate might vary and modeled in function of relative distance. These will give a probability map of availabilities of the navigation mode with landmark detection: \( p(LM|c_i) \). Likewise, availability of the sensor-based relative guidance (e.g. wall following) is conditioned by a limited sensing range with respect to an object-of-interest (wall). Its availability map as a set of probabilities \( p(Wall|c_i) \) is supposed to be given.

These maps and probabilities will be used as a part of the state transition model in the MOMDP planning model. They serve for determining a state transition probability as well for updating the state distribution function conditioned by an observation made on the sensor availabilities.

### 3 Planning model

The goal is to propose a decision framework that will calculate the safest and the shortest path. To achieve this goal, it is necessary to have a GNC system which give a good estimate of the state of the
The MOMDP is an extension for the POMDP model recently proposed by [2] and [10], which explores state can be factorized. Indeed, the state is partially observable but some state variables are known. This model detailed, it is possible to define the model of the planner.

3.2 Recall on MOMDPs

A MOMDP is a tuple \( (s_v, s_c, A, \mathcal{O}, T, R, b_0, \gamma) \), where:

- \( s_v \) is the bounded set of fully observable state variables;
- \( s_c \) is the bounded set of partially observable state variables;
- \( A \) is a bounded set of actions;
- \( \mathcal{O} \) is a bounded set of observation;
- \( T(s, a, s') = T(s_v, s_c, a, s'_v, s'_c) \to [0, 1] \) a transition function.

This work is about the computing of a policy which give us at each epoch the action to execute. Therefore, planning can be associated to decision-making. Decision-making is the cognitive process of choosing which action execute confronted to a situation. Decision-making in real life problem is often synonymous of uncertainty resulting of the stochastic dynamics of the agents (here an UAV) and the environment. Thus this problem can be seen as a sequential decision problem, because these problems are charactized by the enrolment of the problem over the time and that each decision lead to uncertain consequences. POMDPs and variants provide several frameworks to model sequential decision problem under uncertainty and partial observability. The idea behind the POMDPs is that the state is not known but for each state several observations are possible with a specific probability. And the agent when being in a state will receive an observation and update his belief on his state. The Mixed Observability Markov Decision Process (MOMDP) is a variant of the POMDPs, where the state can be factorized. Indeed the state is partially observable but some state variables are known at each epoch. In this problem, the UAV always know the sensor available which can be considered as part of the state, consequently MOMDP can be adapted to the problem.

3.1 Why MOMDP?

This model detailled, it is possible to define the model of the planner.
\begin{itemize}
  \item $O(s', a, o) \rightarrow [0; 1]$ is an observation function such as $O(s', a, o) = O(s'_v, s'_c, a_v, o_v)$ where $p(o_v, o'_v | a_v, s'_v, s'_c) = p(o_v | a_v, s'_v, s'_c) p(o'_v | a_v, s'_c)$;
  \item $R : S_v \times S_c \times A \rightarrow \mathbb{R}$ is a reward function associated with a state-action pair; and:
  \item $b_0 = (s'_0, b, o_0)$, where $b_{0,0}$ is the initial probability distribution over the partially observable states, conditioned on $s'_0$, the fully observable initial states.
  \item $\gamma \in [0, 1]$ is the discount factor.
\end{itemize}

Note, as the probability distribution over states concerns only the $S_v$ set and the observation set $O_v$, the belief state update is redefined as:

$$b_{0,0} \in \alpha (s'_c) = \eta \sum_{s_c \in S_c} p(o_v | s'_v, b, o_v) p(s'_v | s_v, a, s'_c) p(s'_c | s_v, s_c, a) b_{0,0}(s_c)$$

where, $\eta$ is a normalization constant. The belief state $b_c$ is now noted by the couple $(s_v, b_c)$, and $B_c$ is the belief space $s_c$ conditioned by $s_v : B_c(s_v) = \{(s_v, b_c) | b_c \in B_c\}$. $B_c(s_v)$ is a sub-space of $B$, such that $B = \bigcup_{s_v \in S_v} B_c(s_v)$.

Solving MOMDP's consists in finding a set of policies $\pi_{s_v} : B_c \rightarrow A$, which maximize the criterion :

$$\pi_{s_v}^* \leftarrow \arg \max_{\pi_{s_v} \in \pi} E_{\pi_{s_v}} \left[ \sum_{t=0}^{\infty} \gamma^t r((s'_v, b'_c), \pi((s'_v, b'_c))) \right] b_0 = (s'_0, b, o_0)$$

As for the POMDP, the value function at a time step $n < \infty$ can be also represented by a set of $\alpha$-vectors:

$$V_n(s_v, b_c) = \max_{\alpha \in \Gamma_{s_v}} (\alpha \cdot b_c)$$

where $\alpha$ is the hyperplan over the space $B_c(s_v)$. In this way, the value function over the complete state space is parametrized by the set $\Gamma_{s_v}$, i.e. $\Gamma = \Gamma_{s_v}, s_v \in S$. So, given a belief state $(s_v, b_c)$ the optimal action is defined by the action associated with the $\alpha$-vector that maximizes $\max_{\alpha \in \Gamma_{s_v}} (\alpha \cdot b_c)$. For more details about MOMDP algorithm resolution, please see [2].

3.3 MOMDP planning model

Considering the differences between the problem presented and a standard problem from the literature, we propose a model inspired by the MMDP. Some modifications will be made to the MMDP formalism (Figure : 4b). Moreover, the planner model will be based on the GNC model to create the best policy possible.

Our model is defined as a tuple $\{S_v, S_c, A, \Omega, T, O, \mathcal{C}, b_0\}$ :

\begin{itemize}
  \item $S_v$ : bounded set of totally observable states.
  \item $S_c$ : bounded set of non observable states.
  \item $A$ : bounded set of actions.
  \item $\Omega$ : bounded set of observations.
  \item $T$ : The state transition function composed of two functions:
    \begin{itemize}
      \item $T_{S_v} : S_v \times A \rightarrow S_v$ a transition function such as : $T_{S_v}(s_v, a, s'_v) = p(s'_v | s_v, a)$.
      \item $T_{S_c} : S_c \times S_c \rightarrow [0; 1]$ a transition function such as : $T_{S_c}(s'_v, s'_c) = p(s'_c | s'_v)$;
    \end{itemize}
  \item $O : \Omega \times S_v \rightarrow [0; 1]$ : observation function such as :
    $$O(o, a, s'_v, s'_c) = p(o | s'_v, s'_c, a) = \begin{cases} 1 & a = s'_v \\ 0 & \text{otherwise} \end{cases}$$
  \item $\mathcal{C} : \mathcal{B} \times \mathcal{B} \times A \rightarrow \mathbb{R}$ : the cost function, where $\mathcal{B}$ are the belief state space defined on $S$.
  \item $b_0 = (s'_0, b^0_0)$, $b^0_0 \in \mathcal{B}_v$ is the intiale probability distribution over the non observable states, conditioned to $s'_0 \in S_v$, the intiale totally observable state.
\end{itemize}
Note: The set of observations $\Omega$ is equal to $S_v$ and consequently the observation function $O$ is deterministic since $p(o|s'_v, s'_c, a) = 1$ regardless of $s'_v$, $s'_c$ and $a$, since $o = s'_v$. Moreover, the Bayesian dependency in our model changes from a MOMDP proposed in [2] such as $s'_v$ depends on $s'_c$ and $s'_c$ depends on $s_v$ and $s_c$, therefore it depends on the position of the vehicle in the environment.

Figure 4: Difference between the transition model

### 3.3.1 State space of the decisional problem

In our problem, a state $s$ is composed of a set of state variables in two categories: the first being the fully observables state variables $s_v$ and the second being the non observable state variables $s_c$. The state space $S$ represent all the states such as $|S| = |S_v| \times |S_c|$, where $S_v$ represent all the value possible of the state variables $s_v$ and $S_c$ represent all the value possible of the state variables $s_c$. It must noted that for our model, it is the entire state space $S$: $S_v \times S_c$ that are partially observable and we are choosing to factorise according to fully observable and non observable state variables. More specifically, we define $s_c$ such as: $s_c = [X^T \ Y^T \ b_a]^T$. This is the same definition that previously presented in the state transition model. It is necessary to keep the velocity and the bias for the transition function (that will be presented after), because it contributes to compute the next position. $X^T = [x \ y \ z]^T$ is the vector corresponding to the vehicle position, that is defined on a non observable continuous bounded space.

As well as $s_v =$ 

\[
\begin{bmatrix}
\text{GPS} \\
\text{Optical Flow} \\
\text{Landmark} \\
\text{WallFollowing} \\
\text{Collision} \\
P
\end{bmatrix}
\]

is the vector containing the totally observable boolean availability $[0; 1]$ of the sensors and guidance modes, a boolean on the collision, as well as $P$ the localization error covariance matrix.

### 3.3.2 Action space of the decisional problem

In contrary to the position $(x, y, z)$ that is in a continuous space, we define a discret action space. This action space is composed by three action variables.

- $d \in D$: are the directions where the vehicle can move. Since we are in a continuous environment with discrete actions, the actions are chosen arbitrarily. Assuming the UAV was in a 3D grid composed of voxels, the UAV would have 26 adjacent voxels and thus $|D| = 26$ directions $d$ available. It will be considered $|D| = 26$ directions $d$ available in our model even if the UAV is in a continuous environment.

- $m_n \in M_n$: designate the different navigation mode available on the vehicle. The navigation mode determine the sensor set that will be used for the localization.
• \(m_g \in \mathcal{M}_g\): designate the guidance mode available on the UAV. The guidance mode define the guidance law that will be used for the movement.

We pose the hypothesis that the UAV possesses the following navigation mode:

\[
\mathcal{M}_n = \{\text{INS only, GPS/INS, Optical flow/INS, Landmark/INS}\}
\]

and the following guidance mode:

\[
\mathcal{M}_g = \{\text{Absolute Guidance, Relative Guidance}\}
\]

Then we define the action \(a = (d, m_n, m_g)\) as a tuple containing a direction \(d\), a navigation mode \(m_n \in \mathcal{M}_n\) and a guidance mode \(m_g \in \mathcal{M}_g\). According to the number of directions, navigation mode and guidance mode there are at most 208 actions available. Note that the navigation and guidance modes depend of the sensors currently available at a time \(t\) and consequently reduce the number of action possible.

### 3.3.3 Observation space and observation function of the decision problem

As explained, the set of observation \(\Omega\) is considered equal to \(S_v\) and consequently the observation function is deterministic since \(p(o|s'_c, s'_v, a) = 1\) regardless of \(s'_c, s'_v\) and \(a\) as \(o = s'_v\). In contrary to the standard case of POMDP, where \(\Omega\) is the set of all observation that the UAV could receive and give it partial information about the state \(s_c\). In our case, the UAV do not receive any observation in the classical meaning.

**Note:** Considering the navigation error propagation – by using a Kalman Filter –, the drone will receive an inaccuracy measurement on the position due to the sensor. This measurement are used to have this or that measure. In this sense, and in the decision problem, \(s_c\) is considered as a non observable state variable.

### 3.3.4 Transition function

Our transition function between our states is composed of two functions:

- \(T_{s_c}: S_c \times S_v \times A \times S_c \rightarrow [0; 1]\) a transition function such as:

\[
T_{s_c}(s_c, s_v, a, s'_c) = f_{s'_c|s_c,s_v,a}(s'_c|s_c, s_v, a) \sim N(s'_c, \Sigma(s_v))
\]

As previously developed (Eq : 18) \(N(s'_c, \Sigma(s_v))\) is a normal distribution, which designate the probability of a predicted state \(s'_c\), is function of the previous state \(s_c\), of the action \(a\) and the noise \(v\) (Eq : 2) due to the dynamic system. The next state \(s'_c\) depends of the sensors because the sensor-based relative guidance mode require informations from the onboard sensor used during the transition (Eq 15). Moreover, this transition function correspond to process in the motion model (Fig: 1).

- \(T_{s_v}: S_v \times S_c \rightarrow [0; 1]\) is a transition function such as:

\[
T_{s_v}(s'_c, s'_v) = p(s'_v|s'_c)
\]

The function \(T_{s_v}(s'_c, s'_v) = p(s'_v|s'_c)\) represent the transition to \(s'_v\). This transition function depend of the sensor availability map (as indicated in figure 1) and therefore depends only on the next state \(s'_v\). Concretely \(p(s'_v|s'_c)\) is a product of probability on the availability (or not) for each sensor. Which give us \(p(s'_v|s'_c) = \prod_{i=1}^{\left|S_v \setminus P\right|} p(s'_v(i)|s'_c)\) where \(\left|S_v \setminus P\right|\) is the number of sensor on the vehicle and \(s'_v(i)\) is a sensor of the vehicle. Noting that \(P\) is not included in the calculation of \(T_{s_v}(s'_c, s'_v)\) and in this particular case \(s'_c\) represent only the position \(X\).

Then we can tighten the belief state by the probability on the sensor availability. In this intention, we define

\[
T(s_c, s'_v, a, s'_c, s_v) = p(s'_v|s'_c, s_v, a) = p(s'_v|s'_c)f_{s'_c|s_c,s_v,a}(s'_c|s_c, s_v, a)
\]
3.3.5 Belief state

The belief state condenses all the accumulated informations during the path of length $N$, which is the complete information history $h$ defined by:

$$h = \{a_0, o_1, a_1, o_1, ..., a_{N-1}, o_N\}$$

A belief state $b$ can be seen as the probability density function over the possible states at each time step $t$, $b(s^t) = f_s(s = s^t), \forall s^t \in S$. This belief state can be updated after each action $a$ done and at each observation $o$ perceived using the Bayes rule. Thereby, it respects the Markov propriety, since a belief state at the instant $t$ only depends in the belief state at the instant $t-1$, the action done at $t-1$ and the observation observed at $t$. Thus, we must define properly the belief update of our problem, based on the transition and observation function defined before and the previous belief state.

In this approach, it is considered only the reachable belief from the original belief $b^0$. The belief state is a probability density function calculated from the prediction and updated with the sensor available. The first belief state $b^0$ is a confidence ellipse, based on a normal function with the average state and the initial localization error, which has been updated with the sensor available $s_v$. However it is true only for the first belief, because due to the update with the sensor availability map of our model, the belief will not remain a normal function and in our case, it will be represented by a probability density function. By factoring the state space according to our model we obtain:

$$b = (s_v, b_{s_c}), \text{ with } b_{s_c} = f_{s_c|s_v}(s_c|s_v) \text{ and } b_{s_c}^0 = N(\bar{s}_c^0, P_0)$$

3.3.6 Belief state update

After each action we must update the belief state. We join this update with the transition function previously explained (Section: 3.3.4), which depend of an Extended Kalman Filter (EKF). This belief update correspond to the architecture defined previously and respresented by the figure 1. The update is done in two steps based on the transition function and consequently on the three sub-functions:

1. From a belief state $b_{s_c}$ and an action $a$, we will use the state propagation to predict the futur belief state named $b_{s_c}'$. To make the parallel with the diagram of the architecture, this step correspond to the system’s dynamic model which take in inputs $b_{s_c}$ and $a$ and return a new partial belief $b_{s_c}'$.

   $$b_{s_c}'(s'_c) = f_{s_c'|b,a}(s'_c|b,a) = \int f_{s_c'|b,a}(s'_c|s_c, s_v, a)b_{s_c}(s_c) ds_c$$

2. Then, we can calculate the probability to obtain $s'_v$ based on the belief $b_{s_c}'$.
\[ p(s'_c|b, a) = \int p(s'_c|s_c') f_{s'_c|x_c,s_v,a}(s'_c|s_c, s_v, a)b_{s_c}(s_c)ds_c ds'_c \]
\[ = \int p(s'_c|s_c') f_{s'_c|b,a}(s'_c|b,a)ds'_c \]
\[ = \sum_{i=0}^{\left|G\right|} p(s'_c|s'_c \in c_i) f_{s'_c|b,a}(s'_c|b,a)ds'_c \]
\[ = \sum_{i=0}^{\left|G\right|} p(s'_c|s'_c \in c_i)p(s'_c \in c_i|b,a) \]

where \( c_i \) corresponding to a \( i^{th} \) cell of the sensor availability map and \( \left|G\right| \) is the number of cell in the map.

3. The final belief update step correspond to the "Belief state update" of the architecture (Fig: 1). The final \( b_{s'_c|a}^{s_c} \) is compute in function of \( s'_c \), the completely observable state. Remember that this update depends on the real a priori information of the sensor availability map.

\[ b_{s'_c|a}^{s_c}(s'_c) = f_{s'_c|b,a,s'_c}(s'_c|b,a,s'_c) = \frac{p(s'_c|s'_c)b_{s'_c}(s'_c)}{p(s'_c|b,a)} \]
\[ = \frac{p(s'_c|s'_c)b_{s'_c}(s'_c)}{\int p(s'_c|s'_c)b_{s'_c}(s'_c)ds'_c} \]
\[ = \frac{p(s'_c|s'_c)f_{s'_c|b,a}(s'_c|b,a)ds'_c}{\sum_{i=0}^{\left|G\right|} p(s'_c|s'_c \in c_i)p(s'_c \in c_i|b,a)} \]

Therefore the belief state updated is defined as \( b_{s'_c|a}^{s_c} \) and is derived as follow :

\[ b_{s'_c|a}^{s_c} = (s'_c, b_{s'_c|a}^{s_c}) \]  

**Note:** When we write \( f_{s'_c|b,a}(s'_c|b,a) \) it is a misuse of notation, indeed in this case \( b \) is a state probability distribution. In POMDP, the belief state is a complete information state that gather all actions performed and observations received and the intiale state distribution. Thus, when we write in function of \( b \), we write in function of all the past.

### 3.3.7 Cost function of the model

**First cost function** We must remember that the objective is to find the shortest and safest path. To avoid prioritizing neither the safety nor the length in artificial way, the uncertainty corridor was previously introduced in [16] and briefly described next. Intuitively the corridor is created by a sequence of confidence ellipses. This way, more important is the uncertainty during the path, larger the ellipses will be and consequently the volume of the corridor depends directly of the length of the path and of the dispersion of the uncertainty. Our cost function will be based on this uncertainty corridor between two state \( s = (s_v, s_c) \) and \( s' = (s'_v, s'_c) \) knowing that the uncertainty is characterized by \( P \) which is contained in \( s_v \).

Formally, the cost between two state can be defined as :

\[ C(s, s') = \frac{\pi}{6}\|s_c - s'_c\| \cdot (u2_x u3_x + u2_y u3_y + 2(u2_x u3_x + u2_y u3_y)) \]  

(24)
that takes as input a belief state $b$ and will allow it to accomplish the mission. It is necessary to calculate the policy

The goal of the agent (in our case an UAV) is to choose, depending on the situations, the actions which can be considered as an approximation of the first cost function.

3.3.8 Value function

and can be considered as an approximation of the first cost function. In the previous cost function, we calculate the expected cost (volume of the uncertainty corridor) between two belief states. This expected cost is not easily computable due to the double integral. Consequently we propose a second cost function which is based on the uncertainty corridor too. But in the contrary to the first function we calculate the cost between the two belief states by computing the uncertainty corridor between the two averaged states.

The cost between two states do not change (see Eq: 24), but the cost between the belief changes:

$$C(b, b_a^{s'}) = \frac{\pi}{6} \cdot \| \bar{s}_c - \bar{s}_c' \| \cdot (u_{2x'} u_{3x} + u_{2x} u_{3x'} + 2 (u_{2x} u_{3x} + u_{2x'} u_{3x'})) = C(\bar{s}, \bar{s}')$$

Same as before our function $est$ defined by $C : S_e \times S_c \rightarrow \mathbb{R}$. This cost function is easy to calculate and can be considered as an approximation of the first cost function.

3.3.8 Value function

The goal of the agent (in our case an UAV) is to choose, depending on the situations, the actions which will allow it to accomplish the mission. It is necessary to calculate the policy $\pi(b)$, which is a function that takes as input a belief state $b$ and return the action $a$ to be executed such as $\pi(b) \rightarrow a$. This
policy is said deterministic and Markovian because there is only one action to each \( b \). We distinguish
two types of policies, those that have a finite horizon and those that have an infinite horizon. For the
finite horizon policies, we must define an horizon \( N \), at which once the algorithm reaches the policies
will be return (that can be seen as a tree). In our case we are interested in the infinite horizon policies.
They must possess an horizon far enough such as the returned policies is stationary (which mean
that the expected value do not change for a given \( \epsilon \)) and that the expected value of the sum of the
rewards (or costs) be maximized (respectively minimized).

We define the value function \( V^\pi(b) \) as being the expected total cost received from starting in \( b_0 \)
and following the policy \( \pi \). Moreover we choose a discount factor \( \gamma \), \( 0 \leq \gamma < 1 \) that will weight
the expected cost over the time, to ensure the convergence of our policies at an infinite horizon [12].
The value function being based on a sum of costs we need to take into account the particularity of our
function that need the belief state at time \( t+1 \) (see section 3.3.7). As precis our cost do not directly
depend of the action. But the action have a great impact on it because the uncertainty of a state is
function that need the belief state at time \( t \).

Indeed, regardless of the cost function used the entries are the same in our practical case.

We can see that the value function can integrate indifferently the two costs functions that we propose.
Indeed, regardless of the cost function used the entries are the same in our practical case.

\[
V^\pi(b) = \min_{\pi \in \Pi} \mathbb{E} \left[ \sum_{t=0}^{\infty} \gamma^t \mathbb{E} \left[ C(s_t, s_{t+1}) | b_t, b_{t+1}^{\pi(s_{t+1})} = \pi(b_t) \right] | b_0 = b \right]
\]

\[
= \min_{a \in A} \mathbb{E} \left[ C(s_0, s_1) | b_0, b_{1}^{\pi(s_1)} \right] + \mathbb{E} \left[ \min_{\pi \in \Pi} \gamma \mathbb{E} \left[ C(s_t, s_{t+1}) | b_t, b_{t+1}^{\pi(s_{t+1})} \right] | b_{t=0} = b_1 \right] | b_0 = b \right]
\]

So, one can write

\[
V^\pi(b_0) = \min_{a \in A} \sum_{s_{v_1} \in S_v} p(s_{v_1} | b, a)(C(b, b_{1,v_1}^{a}) + \gamma V(b_{1,v_1}^{a})) \tag{26}
\]

We can see that the value function can integrate indifferently the two costs functions that we propose.
Indeed, regardless of the cost function used the entries are the same in our practical case.

4 Discussion and perspective

Having a value function which is PWLC is advantageous, there is an important state of art and the
theories behind it. Furthermore it allow to represent the policy by a set of vector which is easier and
more intuitive. Indeed in contrary to the representation of the policy by a set of all the reachable belief
\( b \in B \), it allow to maintain the policy by keeping a set of vectors. Moreover the theorical work on the
POMDP [13] ensure that if the value function is PWLC then the value function could be approximate
by a PWLC function at each epoch. The algorithms based on this method used this particularity to
accelerate the computing of the optimum policy. Consequently, a major part of the algorithm of the
state of art are based on this representation. In futur work, the value function is not represented by
a set of \( \alpha \)-vector.

The algorithm based on PWLC tend to approach the optimum policy, because the optimum is
often incomputable due to the curse of dimensionality. To approach and guide reasearch of the best
policy, some algorithms use lower and upper bound value functions such as HSVI and SARSOP [14, 9].
This bounds are computed for example using a MDP (in the case of upper bound) that the represent
the solution if problem was complety observable, or using other methods. In this work it is not possible
to use these algorithms. A solution is to use algorithms that are not based on PWLC value functions
like \textit{RTDP-bel} [8] which do not work with \( \alpha \)-vector, but keep an hash-table that maps beliefs to values.
This algorithm coincides with our problem, but the convergence are not proved. Consequently, the
next priority will be to research algorithm that could be adapted or find a new algorithm that can help
to solved the problem. Previously, we developed our model in the perspective to calculate a policy in
a continuous environment. The goal to work on continuous environment and thus continuous state
is to avoid approximation in the planning. In our model, we do not have continuous action which
leads us to important constraints during the path planning. Therefore one important idea we want to incorporate in the future in our model is the continuous actions, which will correspond more to a real application. Continuous actions are not in our first model due to the complexity of the resolution of MOMDP in continuous environment and the complexity induced by continuous actions. Indeed, POMDP with continuous actions are solved with techniques such as particle filter this is why it will be studied later [11].

Références


