2018 IEEE International Conference on Robotics and Automation (ICRA)
May 21-25, 2018, Brisbane, Australia

BFM: a Scalable and Resource-aware Method for Adaptive Mission Planning of UAVs

Chabha Hireche[1], Catherine Dezan[1], Jean-Philippe Diguet [1] and Luis Mejias[2]

***Abstract*— UAVs must continuously adapt their mission to face unexpected internal or external hazards. This paper proposes a new BFM model (Bayesian Networks built from FMEA tables for MDP). This scalable model offers a modular and comprehensive method to incorporate different types of diagnosis modules based on BN (Bayesian Networks) and FMEA table (Failure Mode and Effects Analysis) to mission specifications expressed as a MDP (Markov Decision Processes). The BFM model implements the complete decision making process that covers both the application configurations at the embedded system level and the mission planning at the UAV level. These decisions are based on the QoS (Quality of Service) of applications, the resource use and the system and sensors health. We demonstrate on a case study for a target tracking mission that the BFM model can interface hazards and applications specifications and can improve the success and quality of the mission. To the best of our knowledge, this is the first proposal of a systematic method that integrates diagnosis modules to MDP model in order to take care of the implementation of embedded applications during a mission.**

I. INTRODUCTION

Autonomous vehicles must make decisions with the objective to achieve a mission in an uncertain environment. However, these decisions are strongly heterogeneous and should encompass four categories of choices. The first one is related to environment hazards, it is the mission itself and includes navigation and actions. The second one deals with internal failures, this is the health of the system which is also composed of disjoint classes including energy, state estimation and computing resources. The third one addresses the choice and the fusion of sensors. Finally, the last one refers to the choice of application algorithms and configuration parameters, which are strongly data (values, volume) dependent and also vary with resource (CPU, memory, I/O) availability. These different categories are intricately linked but usually not considered simultaneously because of: i) the complexity of heterogeneity management, ii) the scalability issue when graphs grow and become error prone and not tractable and iii) the partial view of designers who do not master all the required types of expertise.

The main frontier is between the drone (mission, health) and the embedded system (applications, computing resources) levels, which are rarely jointly considered. The drone-level decision is usually not aware of the resources used by the applications. On the other hand self-adaptive approaches in the domain of embedded systems is mission agnostic and so do not consider external conditions (wind, luminosity, temperature) and application demand based on mission phases and action choices. Such methods usually focus on the optimization of performances considering resource and energy constraints [8]. Few works really consider the link between the application quality-of-service (QoS) and hardware/software resource allocation except for specific domains like video coding [3] or motion planning in the robotic field [6].

In this paper, we propose a scalable method to globally manage the different levels of decision-making based on separation of concerns so that each type of experts (Drone mission, Health Management, Embedded System/Application) can focus on their skill domains with comprehensive data. It results in three main contributions. The first one is a modular and scalable method to address the global decision-making process. It is based on a Markov Decision Process (MDP) at the mission level, which is powered by multiple and dedicated Bayesian Networks (BN). The second one is the use of BN based on extended *Failure Mode and Effects Analysis* (FMEA) tables to compute probabilities of solutions including application versions and parameters. They both contribute to the BFM model. The last one is the validation of the proposed model with different scenarios of a UAV performing target tracking.

In the following, we present in Sec. II the usual MDP model used in the state of the art for mission management. Then, we introduce in Sec.III, our BFM approach that uses BN for resource-aware decision and feeds a mission level MDP. In Sec. IV we apply it to a UAV case study with different scenarios. Finally, in Sec.V, we present the results before we conclude and introduce future work.

II. MDP MODEL FOR UAV MISSION

In this section, we first formulate the UAV mission as a MDP problem. Then, we present our approach to extend the MDP UAV mission by including the variations of the environment.

A. MDP Model

A MDP is a probabilistic model for decision making based on markov chains, which consists of finding the action to execute to switch from state *S(i)* to state *S(i+1)* in order to maximize a reward function. The MDP is a quintuplet
$<$ S, A, T, R$>$ [1] where:

- *S*: represents the set of system states.
- *A*: represents the set of possible actions.

* This work was supported by the HPeC project funded by the French National Research Agency (ANR). HPeC project, Nb.15-CE24-0022-01.

[1] Lab-STICC UBO/CNRS, Brest/Lorient, France *Chabha.Hireche@univ-brest.fr, Catherine.Dezan@univ-brest.fr, Jean-Philippe.diguet@univ-ubs.fr*
[2] QUT, Brisbane, Australia *Luis.Mejias@qut.edu.au*

978-1-5386-3081-5/18/$31.00 ©2018 IEEE

- *T*: is the transition function defined on *S*x*A*x*S*. Given the current system state *S(i)* and the action *A*, the probability of the next system state being *S(i+1)* is $P[S(i+1)|S(i), A(i)]$.
- *R*: is the reward function defined on *S*x*A*. It indicates the reward obtained when the action *A* is chosen from the current state *S*.

We illustrate the MDP model in Fig. 1 with a simple tracking mission. In this mission, the UAV flies from the start point to the target area by following a given trajectory, once at the destination area, the UAV is hovering over the area in order to detect the target and to track it.

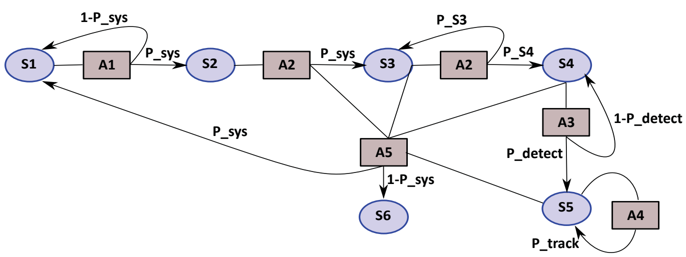

Fig. 1. MDP for tracking mission

1) *System states (S)*: refer to the mission states which represent the different waypoints covered during the mission.
   - *S1*: ground waypoint (starting point).
   - *S2*: first waypoint in flight.
   - *S3*: set of waypoints between the starting point and the tracking area (arrival area).
   - *S4*: represents the detection target state.
   - *S5*: represents the tracking target state.
   - *S6*: is the landing state.

2) *Action set (A)*: represents the different actions that will be executed to ensure the mission.
   - *A1*: is the take-off action.
   - *A2*: represents the "follow trajectory" action.
   - *A3*: indicates the action to detect the target.
   - *A4*: indicates the action to track the target.
   - *A5*: is the action to return back to the base.

3) *Transition functions (T)*: use the following probabilities:
   - *P_sys*: is the probability of good health of the system including the estimation of battery level.
   - *P_S3*: probability to stay in S3 and to follow the trajectory defined by all the intermediate waypoints.
   - *P_S4*: probability to leave S3, it means that the next waypoint is the final one of the intermediate waypoints.
   - *P_detect*: is the probability of the target be detected.
   - *P_track*: is the probability of having a good tracking, in terms of QoS.

   In our approach these probabilities are computed by the dedicated BN as detailed in Sec. III.

4) *Reward functions (R)*: the rewards are fixed according to the priority of the actions during the mission and oriented by the probability values.

B. Positioning

Many previous works consider the uncertainties of the environment to manage a UAV mission [10]. POMDP (Partial Observable Markov Decision Process) is for instance an efficient model that is used to solve the problem of mission management by taking efficient decisions during the mission. A POMDP approach includes everything in a single model, such as the decision related to motion control, mission phase (obstacle avoidance, tracking based on detection event), essentially in the case of localization and orientation [2] [11] [12]. This approach offers good results, however it is neither tractable nor scalable if we involve all decision making categories including algorithm versions and optimization of computing resources.

Our approach aims at separation of concerns. The first level deals with mission phases (navigation, tracking, obstacle detection, etc) and can be specified by the mission designer. The second level includes different modules of diagnosis based on BN, they are organized in three categories: health of the system (resources, battery, etc), the health (QoS) of applications and the health of sensors.

Fig. 2 shows the architecture of the proposed approach. Our model separates the diagnosis module and the decision module. The diagnosis module computes the health states of the different Hardware/Software components of the system (sensors, system and applications). They are given as probabilities considering the context hazards and the internal events of the applications. Different types of Health Management (HM) are then available to feed the decision module (HM for applications, HM for sensors, HM for system). The "Application HM" contain multiples HM, one for each application that can run on the embedded system (e.g. tracking, navigation, obstacle avoidance, etc). The "Sensor HM" also contains different HM for the different sensors or set of sensors (e.g. GPS, camera, etc). Finally the "System HM" includes the HM of the other components of the system as resources (i.e CPU load, etc), battery, etc. The sensor health and system health can impact the QoS of the application. The different HM modules are elaborated using the Bayesian Network model.

The decision module is based on the MDP model and takes as inputs the probabilities computed by the different HM of the diagnosis module. These probabilities, attached to the results of actions, are used in the transition matrices of the MDP.

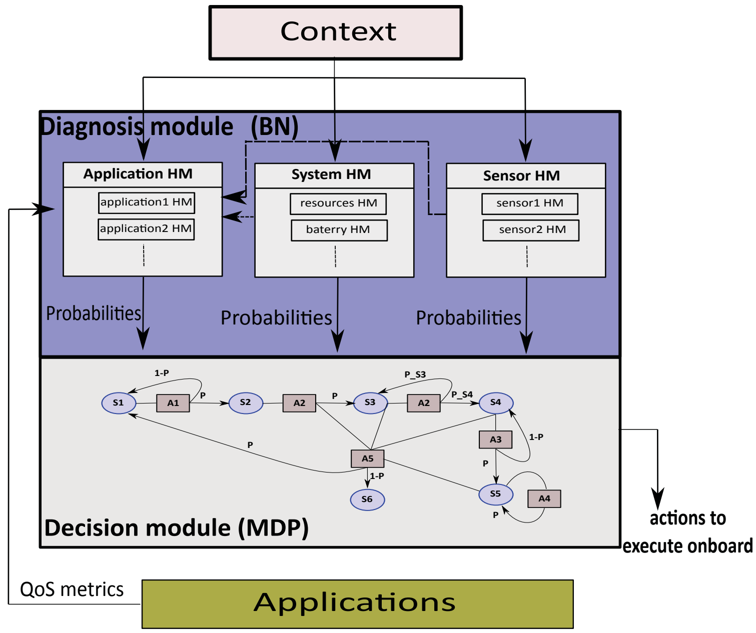

Fig. 2. BFM model architecture for mission management

III. Bayesian Network for resource-aware decision

A. BN model for diagnosis

BN is a probabilistic model used to evaluate the health status of the system by detecting errors that can be observed in a certain context. The nodes of a BN represent random variables, the edge between two nodes represents a conditional dependency. Each node of the network has a probability table (CPT), which indicates the conditional probabilities.

A simple example of BN is given in Fig. 3. The node Tracking (U_Track) represents the QoS tracking (good or bad) considering the context. The Vibration (S_V) and Luminosity (S_L) nodes represent the sensor's observations of whether the QoS tracking is good or not. The CPT entries are fixed on the basis of knowledge of a system's components and their interactions. We can also set the CPT values by learning the parameters of the BN [9].

If we observe a context with "luminosity=low" (put evidence on the luminosity node), then the probability obtained of the QoS tracking is "good" at 10%. The computation of these probability is given by inference.

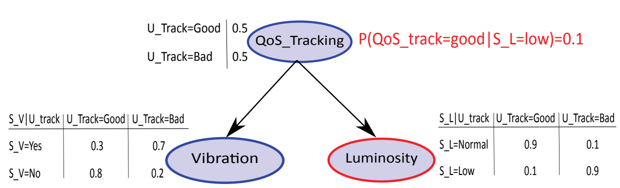

Fig. 3. Simple example of BN

The BN model for diagnosis takes into account the uncertainty of the mission context. Thus, the elaboration of the BN model is not an easy task. For this, we need to define the Failure Mode and Effects Analysis (FMEA) table. This table contain the main errors context with the possible monitoring.

B. FMEA for variants

During UAV missions, evaluation of health status of the components of the system (i.e sensors, actuators, etc) is necessary [13]. On the other hand, the QoS status of the applications which are executed on board, such as tracking application, can also be monitored to ensure mission success.

In this section, we propose that failure mode and effects analysis (FMEA) tables can be used for the tracking task based on the context monitoring. The FMEA table contains the possible errors that can decrease the QoS of the tracking application considering the context. Depending on environment factors, different parameters or versions of the tracking algorithm can be considered to keep good QoS, as shown in Table I.

TABLE I

FMEA applied to the Tracking Application

Errors	**Possible monitoring**	**Appearances context**	**Solution (algorithms)**
Vibration	IMU Vibration sensor	Wind Vibration	Activate the stabilization
Tracking point lost	Model based on: number of features detected (Harris) [5]	Drone speed variations of luminosity	Improve the contrast
Motion vector lost	Model based on: motion vector between 2 images [5]	Target speed Small R.O.I	Raise the R.O.I size

C. From FMEA to BN

We propose to automatically build the BN model from the FMEA table, in order to produce probabilistic diagnosis and decide on the more efficient solution to apply. This model represents the causal relationships and the behavior of the components of the system (hardware/software).

Table II groups the results of this analysis, each U_E_i represents the error that can be observed on the applications or sensors. This error type can be monitored by sensors S_E_i and/or appearances context A_E_i. In our case, we add the column solutions C_E_i which are the solutions adopted to correct the observed error and to maintain the expected QoS of applications.

TABLE II
GENERIC FMEA TABLE

Errors	Possible monitoring	Appearances context	Solution (Applications)
U_E_i	S_E_i	A_E_i	C_E_i

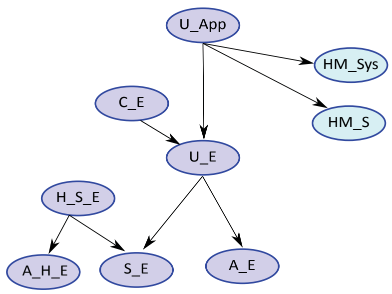

Fig. 4. Generic BN from FMEA

Table II can be translated into a BN model. The resulting BN is shown in Fig. 4 and the definition of the different nodes are the following:

- **Node U_App**: represents the QoS of the application.
- **Node U_E**: represents the unobservable status of the errors.
- **Node S_E**: indicates the measurements by physical or software sensors of error.
- **Node H_S_E**: represents the health state of the error monitor.
- **Node A_E**: indicates the appearances context of the errors, or of the health (**A_H_E**) of the error monitor.
- **Node C_E**: is a command node which indicates the possible solution to activate when the context of error is observed.

We can also add to this model, the observation about the health state of sensors (**HM_S**), which is directly related to the application and of the system (**HM_Sys**) (resource occupancy).

Fig. 5 shows the translation of the FMEA Table I of the tracking application into BN model. The root node represents the QoS of the tracking application that we want to maintain when an error context occurs. The **U_nodes** indicate the unobservable state of the errors of vibrations (**U_Vibration**), motion vector (**U_Loss motion vector**) and error of point tracking (**U_Loss point tracking**). These types of errors are monitored by physical (sensors) or software measurements denoted by **S_nodes** as (**S_IMU**, **S_Model**, ...) and appearances context. Monitor nodes can also have a health status indicated by the **H_S_node** in a certain appearance context **A_H_nodes**. The **C_nodes** represent the solutions that can correct the observed type error (e.g **C_Stabilization** in the case of vibrations error), and ensure to maintain the expected application QoS level.

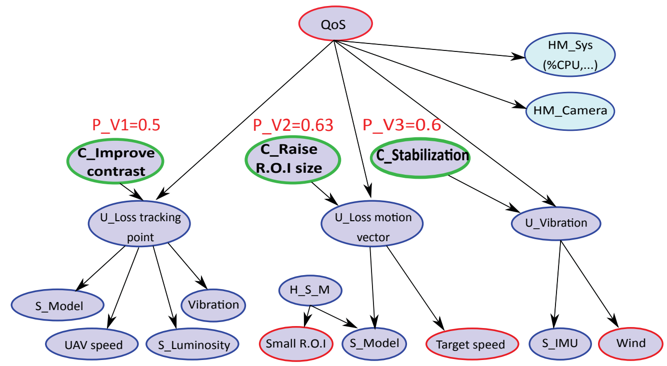

Fig. 5. BN for tracking application from FMEA

Now, we present an example of how to use the BN model. Fig. 5 shows the computation of probabilities associated to solution nodes (green nodes). As mentioned previously, the nodes represent random variables. Each node has two states except the QoS node which has 3 states (high, medium, low).

In this example, we observe a context with wind and speed target and we fix the probability (evidence) of QoS to be in the state "high". The wind introduces some vibrations and tracking error occurs. The target speedup can also lead to a wrong motion vector estimation if the R.O.I (Region Of Interest) is too small regarding the target speed. These observations are reported in the BN by providing evidences on these context nodes, represented by the red nodes in the Fig. 5. We obtain as result, a probability of 60% to activate the stabilization and 63% to activate new tracking version by increasing the window (R.O.I) size. In this example, the efficient solution is to increase the R.O.I size.

IV. ADAPTIVE MISSION USING THE BFM MODEL

A. *Method principles*

In this section, we consider the UAV mission of tracking a target where the UAV flies from the start point to an arrival area by following a given trajectory. Once at the arrival area, the UAV is hovering over this area with the aim to detect a target and track it.

During the mission, errors can occurs due to internal and external factors. In this work, we focus on the tracking phase of the mission and we consider three possible variations from the nominal version (**V0**) for the tracking application (see next subsection).

- **Version 0**: Nominal version of the tracking application using (320x240) frame.

- **Version 1**: The tracking evokes extra stabilization.
- **Version 2**: Histogram-based equalization is added to the reference tracking application to improve the contrast of the image.
- **Version 3**: The tracking version considers a bigger size for the image (640x480 frame) (resize the R.O.I).

The adaptive mission of tracking is modeled using the BFM model. The nodes of the MDP represent the different states of the mission, the squares represent the possibles actions (including the application versions) that can be chosen as shown in Fig. 6.

With the aim to take an efficient decision related on MDP, we need to specify the values of the MDP transition probabilities related to the different versions of the tracking application. These probability values are computed by the BN diagnosis modules as shown in Sec. III. The reward values are defined by the version probability values, If the probability of version 1 for example is higher than the two other versions then its reward is the higher one.

B. Case studies

Taking the adaptive mission of tracking, we elaborate four different scenarios including different versions of the embedded tracking application. These scenarios are as follows:

- **Scenario 1**: nominal case without context errors.
- **Scenario 2**: observation of vibration error through the tracking application BN model
- **Scenario 3**: the target goes faster than the UAV. Thus, there is a high probability of losing the target on the image.
- **Scenario 4**: is an extension of scenario 3 with presence of strong wind. We observe that the battery consumption decreases more rapidly.
  In our case, the estimation of the battery consumption is modeled using DBN (Dynamic Bayesian Network) model, which is a BN unrolled on two steps. This model computes the probability of the remaining energy taking into account wind, temperature and the new application activated [13].

V. Results

In this section, we present the results of our experiments. We used Matlab for simulations using BFM model to different scenarios of the tracking mission.

The MDP of the mission is regularly evaluated and takes as inputs the probabilities provided by the BN diagnosis modules during time. These diagnosis modules are elaborated for the HM of the energy and the HM of the tracking application with the different application versions. We run the MDP during 100 steps with a finite horizon resolution.

We compare the results obtained between the reference mission of tracking and the adaptive mission of tracking (extended version) in terms of success rate, which is measured by the number of time steps for the tracking activity, as shown in Table III. The reference mission is based on the standard MDP implementation described in Fig. 1.

TABLE III
COMPARISON OF THE REFERENCE MODEL AND THE BFM MODEL FOR THE TRACKING MISSION

Scenario	Tracking time (nbr steps)		# Cycles (10^6)
	Reference	BFM	Tracking version
Nominal	51	51	103 (320x240 frame)
Vibrations	40	65	292 (nominal + stabilization)
High Speed	40	72	264 (640x480 frame)
Wind	35	56	264

The results show that the obtained tracking time varies between the different scenarios presented in the previous section:

- **Scenario 1**: this is the nominal case, the tracking time obtained with the BFM model is similar to the one of reference MDP mission. The tracking is done with success for both, if no hazard appears.
- **Scenario 2**: high vibrations are observed during the tracking phase. We can see that the reference mission of tracking is aborted due to these vibrations. However, with the BFM model, we continue the tracking by activating a new version of tracking application with extra stabilization.
- **Scenario 3**: target speed scenario, the target is temporarily lost. With the BFM model, this problem is corrected by resizing the R.O.I of the tracking. As a result, the tracking time with the new MDP mission is longer than the reference mission.
- **Scenario 4**: is an extension of scenario 3 with wind. The adequate solution (resize the R.O.I of tracking) is already activated but extra energy is consumed due to the presence of wind. As a result, the tracking time with the BFM model is still greater than the one of the reference mission.
  In the event that bad luminosity is observed, the BFM model activates tracking version with pre-filtering (version 2 of tracking application) to improve contrast of the image. The result expected is similar to the high speed scenario Sec. IV-B.

The experimentation shows that we can extend the tracking time by considering different possible tracking versions that can correct or decrease the tracking error linked to the context.

On the other hand, we also give the execution time to illustrate the variations of resource use. We consider a usual Cortex A9 embedded processor with a NEON coprocessor and a 925MHz clock. The tracking application is TLD [4] (aka Predator) with two window sizes (small: 320x240 and large: 640x480). The stabilization and landing area detection applications are our own implementations and use OpenCV.

As shown in the Table III, the nominal version can reach about 9 fps (103.10^6 cycles), if it can use all CPU time. In

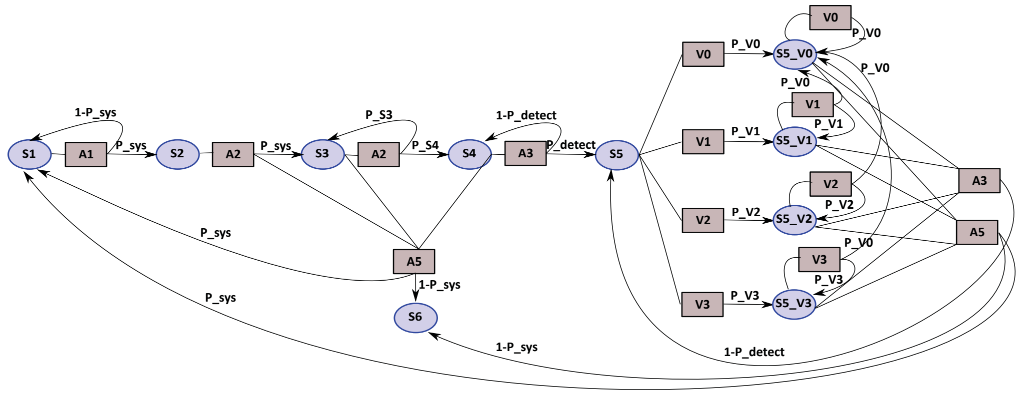

Fig. 6. MDP for tracking version mission

the context of scenario 2, the new tracking version (nominal tracking + stabilization) is slower and needs 292.10^6 cycles. In the case of the high speed scenario, the tracking version adopted (double R.O.I size) is obviously slower and requires 264.10^6 cycles. Thus, all these application versions require an extra number of resources to run on CPU. In consequence, if the extra burden overloads the processor then the expected QoS cannot be delivered. So it must be considered in the global mission management.

As an example of another possible event-based processing load, we consider an important computer-vision task for emergency landing that identify flat safe areas [7]. This implementation task requires, if the battery level is dangerously lower than expected, additional 364.10^6 cycles.

VI. Conclusion

MDP is a common model for mission specification under uncertainties. The main problem with such a model is to find relevant values for the probabilities of the transition functions and the rewards to take into account. We propose in this paper to enhance the classical MDP model with diagnosis modules built from FMEA tables. As the FMEA tables are automatically transformed into Bayesian networks, this new model is called BFM (Bayesian Networks from FMEA for MDP). We show that this new model can help to perform the mission objectives more efficiently by introducing the low level of the application to be executed on board. We take the example of the tracking mission for this demonstration and observe that the tracking with the new mode succeeds (up to 80% better) by applying the task adaptations to face hazards of the mission.

As we can observe in the case study, a usual embedded processor will be overloaded if higher frame rates and multiple applications are required. Moreover, the embedded system must also run other important tasks such as path-planning and obstacle avoidance. So the management of computing resources is a critical question that must be considered as one important dimension of the mission planning. One of the solution is to consider concurrent BFM models, describing each functional phases of the UAV mission as a BFM model (navigation, emergency landing, tracking, etc), which include the estimation of computing resources (FPGA, CPU).

Future work is the adaptation of processing resources according to the mission demands. We will extend the set of application versions by considering dynamic hybrid architecture where FPGA dynamic reconfiguration and GPU provide alternative hardware implementations with higher performances.

References

[1] Christos G Cassandras and Stephane Lafortune. *Introduction to discrete event systems*. Springer Science & Business Media, 2009.

[2] Caroline Ponzoni Carvalho Chanel, Florent Teichteil-Königsbuch, and Charles Lesire. Multi-target detection and recognition by uavs using online pomdps. In *AAAI*, pages 1381–1387, 2013.

[3] Yanjiao Chen, Fan Zhang, Kaishun Wu, and Qian Zhang. Qoe-aware dynamic video rate adaptation. In *IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, 2015.

[4] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-learning-detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 34(7):1409–1422, July 2012.

[5] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Tracking-learning-detection. *IEEE trans. on pattern analysis and machine intelligence*, 34(7):1409–1422, 2012.

[6] Manfred Kröhnert, Raphael Grimm, Nikolaus Vahrenkamp, and Tamim Asfour. Resource-aware motion planning. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 32–39, 2016.

[7] Luis Mejias and Daniel Fitzgerald. A multi-layered approach for site detection in uas emergency landing scenarios using geometry-based image segmentation. In *International Conf. on Unmanned Aircraft Systems (ICUAS)*, pages 366–372, 2013.

[8] Arslan Munir, Ann Gordon-Ross, Susan Lysecky, and Roman Lysecky. Online algorithms for wireless sensor networks dynamic optimization. In *IEEE Consumer Communications and Networking Conference (CCNC)*, pages 180–187, 2012.

[9] Carsten Riggelsen. Learning bayesian networks from incomplete data: An efficient method for generating approximate predictive distributions. In *SIAM, Int. Conf. on Data Mining*, pages 130–140, 2006.

[10] Andrew G Shem, Thomas A Mazzuchi, and Shahram Sarkani. Addressing uncertainty in uav navigation decision-making. *IEEE Transactions on Aerospace and Electronic Systems*, 44(1), 2008.

[11] Fernando Vanegas, Duncan Campbell, Nicholas Roy, Kevin J Gaston, and Felipe Gonzalez. Uav tracking and following a ground target under motion and localisation uncertainty. In *IEEE Aerospace Conference*, pages 1–10, 2017.

[12] Fernando Vanegas and Felipe Gonzalez. Uncertainty based online planning for uav target finding in cluttered and gps-denied environments. In *IEEE Aerospace Conference*, pages 1–9, 2016.

[13] Sara Zermani, Catherine Dezan, Chabha Hireche, Reinhardt Euler, and Jean-Philippe Diguet. Embedded context aware diagnosis for a uav soc platform. *Microprocessors and Microsystems*, 51:185–197, 2017.