Autonomous Surveillance Robots
A Decision-Making Framework for Networked Muiltiagent Systems

S. Witwicki, J.C. Castillo, J. Messias, J. Capitán, F. Melo, P.U. Lima and **M. Veloso**

Abstract

This paper proposes an architecture for an intelligent surveillance system, where the aim is to mitigate the burden on humans in conventional surveillance systems by incorporating intelligent interfaces, computer vision, and autonomous mobile robots. Central to the intelligent surveillance system is the application of research into planning and decision-making in this novel context. We frame the robot surveillance decision problem, describing how the integration of components in our system supports fully-automated decision-making. Several concrete scenarios deployed in real surveillance environments exemplify both the flexibility of our system to experiment with different representations and algorithms *and* the portability of our system into a variety of problem contexts. Moreover, these scenarios demonstrate how planning enables robots to effectively balance surveillance objectives, autonomously performing the job of human patrols and responders.

1 Introduction

Combining recent research advances in computer vision, robot autonomy, and artificial intelligence has the potential to revolutionize surveillance technology. Consider the careful attention spent by security personnel to monitor numerous live video feeds from cameras that are presently surveilling our parking lots, university campuses, and shopping malls. Imagine the monotonous patrols of armies of security guards through countless corridors. Deliberate over the difficult strategic decisions of where and how to allocate precious human resources, both in response to immediate security concerns *and* in anticipation of future conditions. To maintain safety and security, the conventional surveillance system relies critically on human attention, action, and intelligence. However, such reliance is untenable in a society where the trend is for more cameras, embedded in larger and more complex surveillance environments, to fend against a growing array of potential threats (from burglary, to natural disasters, to terrorist attacks). In this paper, we advocate a shift of reliance onto autonomous system components, in order to scale to meet present-day surveillance needs.

One aspect of surveillance that has received considerable attention from researchers is real-time scene analysis. Systems have already been developed to autonomously analyze video streams in environments such as transportation networks [6, 27] and public spaces [5], so as to identify actors and characterize their behavior. Recent examples include IBM's Smart Surveillance System (S3) project [22] and Yao *et al.*'s system for cooperative object tracking [30]. There are also approaches for activity interpretation [8, 12, 13, 20, 25], while other works are more focused on meeting low-bandwidth requirements by locally processing surveillance images [4]. Although these systems can autonomously extract relevant information for surveillance purposes, they are still heavily dependent on a team of human security personnel, for instance to cover areas which may be outside of the range of the stationary sensor network and to resolve situations that may require physical intervention. Our work aims to increase autonomy and to reduce the human burden by introducing autonomous mobile robots into the surveillance system.

Research in *robot mobility* has advanced to the point that robots now have the capability of navigating complex environments, patrolling as humans guards would do. Equipped with cameras and other sensors of their own, they can also serve as mobile surveillance nodes augmenting a network of statically-situated cameras. For instance, a robot can provide temporary coverage to areas that may become critical due to camera failures or other anomalies. Moreover, robots have the mobility, sensors, and actuators to respond directly to events detected over fixed camera streams, thereby leveraging *real-time scene analysis*.

To integrate these complementary research technologies effectively, and to render robots truly autonomous, requires a third key technology: *intelligent decision making*. Robots should choose their actions so as to to fulfill a combination of objectives given limited resources. This is often framed as a (multi-)robot task selection (and allocation) problem [10], and has been approached through a variety of AI techniques: from logic-based (classical) planning methods [9], to market (auction)-based solutions [15] and those based on constraint optimization [16]. An obstacle to applying such techniques here is that surveillance decisions are riddled with *uncertainty*. Uncertainty is present in robots' awareness, due to imperfect sensing and localization, as well as in environmental dynamics, due to imprecise control and unpredictability about when surveillance events may occur. This challenge leads us to examine state-of-the-art formalisms for modeling robots' task dynamics and for planning under un-

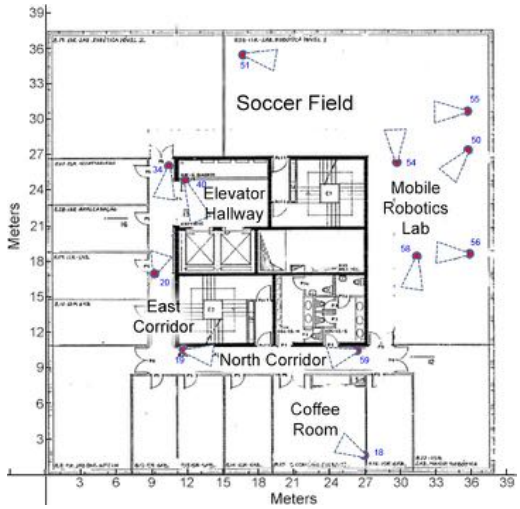

Figure 1: A staged indoor surveillance environment with the positions of the static cameras (red circles) and the common coordinate system for event location.

certainty that push the boundaries of robot autonomy.

The primary contribution of this work, however, is the integration of complementary research technologies from video surveillance, mobile robotics, and AI. We demonstrate the efficacy of our integration through a prototype system that includes a small number of robots and cameras deployed in realistic surveillance environments. A modular architecture and general framework for representing and communicating surveillance events makes our system a useful testbed for experimenting with various research technologies. In contrast to past results in multi-robot surveillance that employ human operators to orchestrate the behavior of the robots [7], we are able to achieve fully autonomous surveillance robots capable of making decisions on their own, with the potential to help human operators.

2 Overview of Surveillance Framework

We begin with a brief overview of our framework, which is motivated by a concrete example of a decision faced by a surveillance robot. This leads us to characterize the decision-making problem, as well as to structure our system in support of the implementation and testing of decision-theoretic planning for mobile surveillance robots.

2.1 Motivating Example

Imagine adding a robot to the surveillance environment shown in Figure 1. In contrast to the static cameras placed at fixed positions, the robot is capable of dynamically patrolling the building. It can move from room to room, using its sensors to scan for anomalies that the static cameras might have missed, and using its actuators to interact with the environment in ways that a static camera cannot. The robot's limitation, however, is that it can only occupy one physical location at a time.

Consider that, late one night, the robot is patrolling the *east corridor* on its way to the *elevator hallway*. Suddenly, one of the fixed cameras detects a person moving in the *north corridor*. At this time of day, the north corridor has restricted access, arousing suspicion that someone is trespassing. Assuming this event is communicated to the robot across the network, the robot could turn around and proceed directly to the detection location. Alternatively, the robot could continue along in order to surveil the elevator hallway, which is also an important room in the building. This example illustrates the kind of relevant decisions that a surveillance robot could face given its current status and the status of the surveillance system. The decision of whether to respond immediately to an event or to continue patrolling should be made carefully and deliberately, since it may compromise the security of the building.

Challenge	Explanation
Constrained resources	A robot has a finite operation time and cannot visit all locations instantaneously.
Urgency / priority	A trespassing event left unaddressed for too long can turn into a robbery.
Uncertainty about event occurrences	It is unknown when, where, and even if an event will occur.
Uncertainty in decision consequences	E.g., there is no guarantee that the robot will succeed in its actions, e.g., thwarting the trespasser.
Uncertainty in the sensorial data	Imperfect detection methods may yield false-positives and -negatives.
Coordination of decisions	A robot team should handle events in parallel, avoiding redundancy.
Intermittent communication	E.g., when robots traverse large and complex spaces with dead zones.

Table 1: Challenges of surveillance decision-making.

2.2 Modular System Design for Decision-making

In general, a mobile surveillance robot will experience a sequence of decisions about where to go and what to do, as long as it is operating in the environment and events are being detected by the network. In order to increase the autonomy of the networked robotic system, planning methodologies should consider several relevant aspects within the decision-making problem, as summarized in Table 1.

In addition to accommodating various decision-making methodologies, an effective autonomous surveillance framework needs to deal with a wide range of heterogeneous sensors and actuators exchanging information in real time: differing robot platforms, lasers, cameras, microphones, speakers, etc. Therefore, we propose here a modular framework for surveillance that decomposes the overall system into components and defines a set of interfaces for component interaction and communication. The system is versatile enough to allow for adaptable reuse as well as the incorporation of new functionalities (e.g., new sensor technologies).

Figure 2 diagrams our modular surveillance framework. Apart from the heterogeneous sensor/actuator modules, a Human-Machine Interaction (HMI) module is included to display information (e.g. detected events) to the operator, to

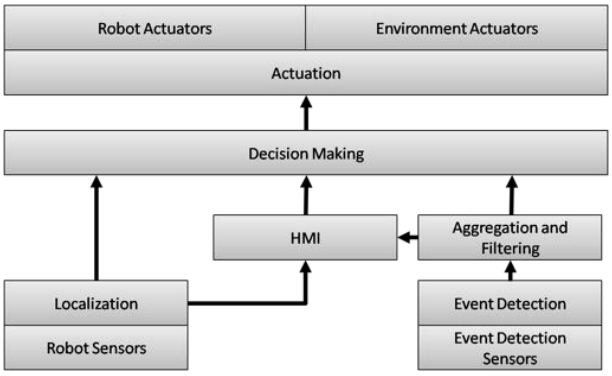

Figure 2: Modular design of our surveillance framework.

receive remote commands (e.g. sending a robot to a desired position), and to produce audible signals from each robot in the form of speech whereby the robot can interact with people in the environment.

3 Detecting and Disseminating Events

Events, such as a person requiring assistance or an intrusion, form the basis for all intelligent surveillance activities. In this section we describe where these events come from and how they are automatically detected and represented in support of effective robot planning. For illustrative purposes, we focus our description on the *trespassing* event introduced in Section 2.1.

3.1 Image Processing

The multi-camera system requires live video acquisition and transmission. High-resolution camera images need to be captured and received at a steady rate and reliably enough to perform event detection. This involves high-bandwidth computation balanced across several high-performance servers, each processing the images in real time.

Our surveillance system integrates the technique proposed in [19] for both detecting people as they are moving around the scenario and for detecting activities or other events (such as a person waving for assistance, trespassing on a forbidden area, etc). Note that other image processing algorithms could be plugged into our system, since the framework is flexible, requiring only that new modules respect the interfaces to communicate with connected modules. The processing is divided into two main phases: (1) Human presence is detected by a background-subtraction-based algorithm; the human is subsequently tracked via data association between consecutive frames. (2) Human activity is detected by means of a classifier that analyzes a tracked person's movements through optical flow computation. Table 2 shows the performance of our algorithm for waving detection compared to some state-of-the-art techniques on the KTH action database[1]. In the method, the temporal support of the classification of every sequence uses an event window size of 4s

[1]KTH action database `http://www.nada.kth.se/cvap/actions/`

Technique	Accuracy
Our method	91.7%
Niebles et al. [20]	93%
Ke et al. (1) [13]	88%
Ke et al. (2) [12]	91.7%
Schuldt et al. [25]	73.6%

Table 2: Accuracy of state-of-the-art methods for waving detection.

(at 25 frames per second), considering a waving event if at least 60% of the single-frame classifications are positive in that sequence. More details and results of our method can be found in [19].

Continuing with our running example, Figure 3(a) highlights two cameras located in the area labeled as *north corridor* with overlapping fields of view. Figure 3(b) illustrates how the detections of a person on the image plane are translated into positions on the global coordinate frame of the scenario (depicted on the axes of Figure 1). This coordinate system is shared by all robots and image coordinates can be translated to it by means of homography-based transformations. Along with the detected positions, we model uncertainties that capture the detection imprecision of the sensor itself (illustrated as ellipses in Figure 3). False positives (the detected event did not actually occur) and false negatives (an event was missed) are thereby modeled probabilistically. Once detected, the events are sent to the *Aggregation and Filtering* block in Figure 2.

3.2 Event Aggregation and Filtering

To mitigate the noisy measurements produced by state-of-the-art image processing algorithms, and to improve the consistency of human detection, we aggregate information from multiple overlapping cameras. In our system, cameras provide events as 3D positions and orientations with their associated uncertainties (modeled as a 6×6 covariance matrix), which are then aggregated together in a centralized fashion. We keep track of the position of every event detected, and once new camera detections are received, data association is used to match detections of previously-detected actors or distinguish detections of new actors. Data association in our multi-camera scenario is solved by methods such as Kullback-Leibler Divergence (KLD) [14].

Figure 3 shows how overlapping cameras can capture detections from the same person that need to be combined by the aggregation module. The aggregation module receives asynchronously detections from multiple cameras and updates the information of the corresponding tracks accordingly (or creates new tracks when required). The event filtering system recognizes the tracked detection as *trespassing* by way of a predefined abstraction of the scenario map wherein some areas are marked as forbidden (Figure 3(c)).

Once it has been detected that a person is trespassing, or other relevant human activity, the system generates and transmits a corresponding meta-event to the mobile robots.

(a) (b) (c)

Figure 3: Running example of trespassing event detection: (a) images acquired by camera 59 (top) and camera 19 (bottom), with detections; (b) (top) field of view and detection of camera 59, (bottom) field of view and detection of camera 19; (c) a scenario abstraction map, where red zones correspond to restricted areas, white zones to accessible areas, and dark zones to areas unsuitable for robot event attendance (e.g. cluttered zones).

4 Autonomous Mobile Robot Responders

To play the part of human security guards, mobile robots should be capable of responding to surveillance events regardless of when, where, and whether they may occur. The random nature of such events induces a problem of decision-making under uncertainty at various levels of abstraction: the robot team should cooperatively decide which robot, if any, should respond to a new event (task allocation); robots should respond to events in the most efficient manner (task execution); and each robot should routinely decide where to position itself in anticipation of an event (navigation). In this section, we describe how the decision-making problems in our surveillance framework are modeled symbolically, enabling their treatment through automated planning and reasoning mechanisms.

4.1 Abstracting the System and its Environment

Consider modeling the autonomous robots' decisions by simulating in detail the many possible detections of events and the various actuations of motors by which each robot could travel to all of the possible event locations. Due to the great deal of continuous variables involved, and the unpredictability of the events, the original optimization problem derived from making low-level decisions may be intractable. In order to tackle this decision-making problem, it is therefore necessary to describe it at a coarser level of abstraction, including only as much information as that which is deemed relevant to differentiate between the outcomes of the possible decisions of the robots.

First, we partition the environment into a discrete set of locations that can be encoded as a topological graph onto which the position of the robots and of the detected events can be mapped. Second, we discretize the space of possible controls for the robots as abstract "movement actions". From each node in the topological graph (describing the location of each robot), there are as many movement actions as adjacent nodes. These actions invoke the robot's lower-level path planner, driving it to a predefined "waypoint" associated with graph node (though those actions may fail, leading to non-deterministic transitions). In particular, we assume that the robots are equipped with on-board sensors for localization and navigation. Standard probabilistic localization methods and path planning algorithms can be used.

The environment of our running example shown in Figure 3, when discretized in the above manner, results in topological graph describing reachable locations depicted in Figure 4. This discrete representation of location is then coupled with additional symbolic variables that impact a robot's decisions, for instance, the type and nature of each detected event (e.g., trespassing). The selection of symbolic variables depends on the desired behavior of the system (as we elaborate in the next section). Moreover, different automated planning mechanisms may expressly depend on different representations of the environment. For instance, while logic-based planners rely on predicate-based representations of these variables, decision-theoretic planners can operate directly over integer-valued discrete representations. The common thread, however, is a discrete representation of the "state" of the system as a whole, and of the decisions (or "actions") that can be performed at the time of each event.

4.2 Formalizing the Decision-Making Problem

Equipped with a symbolic description of the system and of the capabilities of each robot, we can then formalize the decision-making problem. Let $s_t \in \mathcal{S}$ represent the discrete "state" of the system at some discrete time t, which is typically a tuple of symbolic variables as described above. At each time t, the robot(s) must select an "action" $a_t \in \mathcal{A}_t$, where $\mathcal{A}_t$ represents the set of possible symbolic decisions available at that time. The execution of a_t influences the resulting state at the next decision episode, s_{t+1}.

In our running example, one way of modeling the state is $s_t = \langle r_t, x_t^1, \ldots, x_t^6, b_t \rangle$, where r_t represents the topological position of the robot (within the possible alternatives represented in Figure 4); $x^{1,\ldots,6}$ are the states of each topological node, which could be modeled, for instance, as

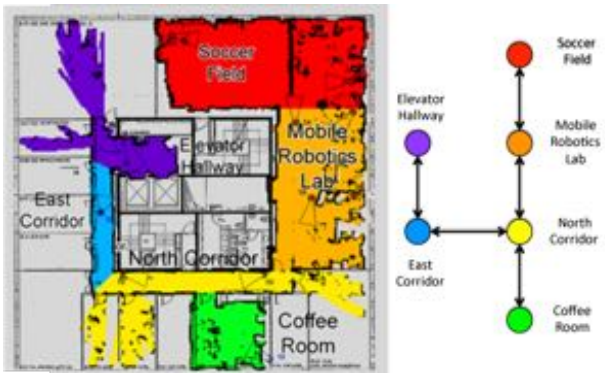

Figure 4: A map of the environment of our running example partitioned into areas of interest overlayed with the laser-based map used for robot navigation (left); the topological graph corresponding to this discretization, and which is used in the decision-making block of our system.

$x^i \in \{\text{'Unknown'}, \text{'Clear'}, \text{'Intruder'}\}$; and b_t represents the battery level of the robot. Additionally, the actions at each time could be the high-level navigation movements between nodes of the topological graph, and also other possible interactions of the robot with its environment, *e.g.*, $\mathcal{A} = \{\text{'Up'}, \text{'Down'}, \text{'Left'}, \text{'Right'}, \text{'Expel Intruder'}\}$.

Given these symbolic representations of states and actions, the general decision-making process can be cast as the following optimization problem: at each time t, given the history of states and actions $\langle s_0, a_0, s_1, a_1 \ldots, s_{t-1}, a_{t-1} \rangle$, select a new action a_t to satisfy one of the following optimization targets:

- (**Either**) maximize a target utility function of future visited states and selected actions (utility-based planning);
- (**Or**) minimize the number of decisions needed to reach a certain goal state (goal-directed planning).

This formulation of the decision-making process is general enough to encompass most logic-based and decision-theoretic planning methodologies.

4.3 Application of Decision-Theoretic Planners

As motivated in the preceding sections, decision-theoretic planning methods are especially applicable to the type of problems involved in our multi-agent surveillance system, since they account for multiple sources of uncertainty in the environment. As such, we have opted to apply these methods to obtain decision-making policies for the robot team in our implementation of the surveillance system.

Most decision-theoretic methods are based on the concept of Markov Decision Processes (MDPs) or its extensions [3]. An MDP is an instantiation of the decision-making process defined in the previous subsection, where the state transitions after executing a team action are modeled with a transition probability function, and the relative priorities of each state and action (desired behavior) are encoded by a reward function.

The objective in an MDP is to obtain a particular mapping of states to actions, $\pi : \mathcal{S} \rightarrow \mathcal{A}$ (a *policy*) that maximizes the *expected* accumulated reward over a certain (possibly infinite) number of future steps (i.e., utility-based planning).

The definition of the components of an MDP is domain-dependent. For instance, in our running example, the transition function depends on the probability that the robot successfully completes its navigation actions, and the probability that an intruder appears in a room. Each time that the robot visits a room, its state changes to either 'Clear' or 'Intruder'. If the robot has not visited a room for some time, its state could be reset to 'Unknown', symbolizing a lack of information regarding its occupancy.

Furthermore, a positive reward could be assigned to a state in which all rooms are known to be 'Clear', and likewise a negative reward to a room that has an 'Intruder'. No reward would be given for 'Unknown' rooms. Since the robot's policy attempts to maximize reward, this would induce the robot to try to visit all rooms as fast as possible (automatically determining an optimal patrol order), while at the same time prioritizing its response to 'Intruder' states. A more specific definition of the transition and reward models for a surveillance task that is analogous to our running example can be found in [29] and in the supplementary material.

In some applications, considering the effect of limited or noisy information may be important for decision-making. Partially Observable MDPs (POMDPs) are an extension of MDPs which also account for uncertainty when observing the state [26], and they are appropriate when the cameras can produce unreliable detections. Although calculating policies for POMDPs is computationally more demanding, we demonstrate in Section 5.3 that this calculation is feasible for a handful of robots, and discuss in Section 5.4 how such models could be scaled to larger autonomous surveillance problems.

5 Case Studies

In the preceding sections, we have illustrated the various aspects of our autonomous robot surveillance framework using a simple running example. We now turn to several concrete case studies, wherein we formulate and solve the decision-making problem using state-of-the-art planning techniques, and deploy the resulting plans in real robots. The case studies involve different environments, events, robot capabilities, and planning algorithms, showcasing the generality of our framework. Specific details on the models used can be found in the supplementary material.

5.1 Common Implementation of Components

With the aim of portability and flexibility, we have implemented our surveillance framework described in Section 2 on top of the widely-adopted ROS infrastructure [24]. Our implementation consists of three macro-blocks communicating by means of ROS *topics* (see Figure 5). First, a "Robot" macro-block is run on each surveillance robot, acting as its intelligence. The modules for robot localization and navigation of our framework described in Figure 2 are here implemented by means of the ROS *Navigation Stack*,

which provides Monte-Carlo localization and standard algorithms to navigate waypoints in a map. Moreover, the Decision-Making module in Figure 2 is here implemented by means of MDP or POMDP planners[2], which will be described later. Those planners are in charge of determining the best action for each robot and sending the corresponding command to the navigation components.

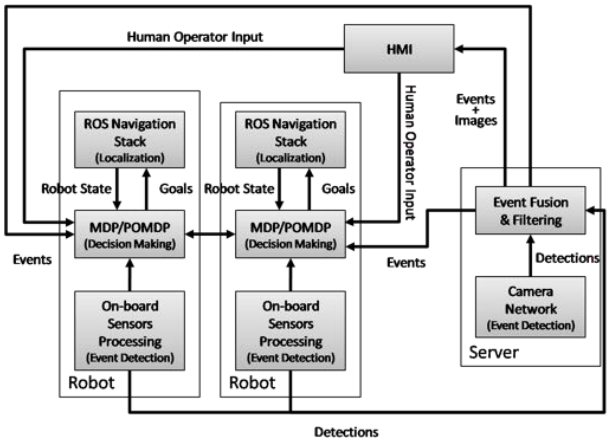

Figure 5: ROS-based implementation of the modules of our autonomous surveillance system with multiple robots.

The "Server" macro-block is in charge of detecting events and is run on one of several physical machines wired to the network. This macro-block receives the image streams from all the cameras (including cameras onboard the robots) and performs the algorithms described in Section 3 to generate events. Those events are communicated to the robots and to the third macro-block, "HMI", which handles all interactions with the human operators. This module is distributed into different applications. Here, we have implemented a central videowall application that allows operators to select image streams from the different cameras. Information about detected events is overlayed onto the images (as in Figure 10). We have also implemented an alternative application for mobile devices (tablets) where the operators can check events. Moreover, by interacting with a videowall displayed on their mobile devices, operators are able to send the robots to specific locations that they consider relevant for surveillance.

In each case, autonomous robot surveillance comprises a subset of the following four types of activities:

Patrol of the environment. The robots should maintain under close surveillance all reachable areas in the environment, paying particular attention to those most sensitive (e.g., with valuable items or not covered by static cameras). Given the dynamic nature of the environment, robots should continue to visit all areas, not neglecting any area for too long, over the course of the entire surveillance mission.

[2]The MDM package: `http://wiki.ros.org/markov_decision_making`

Assistance to visitors. As noted in Section 3, the camera network can automatically detect events related to human activity, for instance, whether a visitor is requesting assistance (by waving to a camera). In response to such an event, one of the robots should meet the visitor, and perform a simple interaction with the intent of aiding the visitor by engaging in a simple dialog and then guiding him or her to whichever the visitor indicates as the desired destination.

Security of restricted areas. Another event related to human activity is triggered whenever a person is detected to be trespassing in a restricted area. In this situation, one of the robots should navigate to the corresponding position of the detection and warn the trespasser, potentially alerting human security to help resolve the situation.

Emergency response. We also consider emergency situations that require an immediate response by the robots. For example, if a fire breaks out in the operating environment, robots can use additional sensors to verify whether or not it was a false alarm, and even to help put out the fire if capable.

5.2 MDPs for Single-Robot Surveillance

In the first set of case studies we apply an MDP technique to control a single robot following the behaviors described above. The MDP formulation is described in Section 4.1, with the robot selecting new actions whenever an event occurs or its position changes. The state space is factored into multiple variables: one for each possible event occurrence in the system (e.g., assistance requests, trespassing situations, emergencies), and one for the position of the robot. The robot's policy is computed using an MDP model whose transition probabilities were inferred from a combination of experimental data and statistical inference, and whose rewards were hand-tuned to balance the objectives. Analytical experiments have shown that the MDP approach remains tractable over long time horizons, though the performance is crucially dependent on the accuracy of (bounded) predictions of event likelihoods. Further details of our surveillance MDP model specification can be found in the supplementary material.

Deployment in a testbed First, we performed experiments in the scenario of Figure 1, which is a surveillance testbed on the floor of our research institute [2] that includes 12 static cameras, three servers, and one Pioneer 3-AT robot. The Pioneer 3-AT was a four wheel drive robot equipped with a SICK laser, a webcam and speakers; programmed to navigate around the scenario, to survey remote events, and to speak warning messages. The map of the scenario together with the corresponding topological map can be seen in Figure 4. Here, a visitor can ask for assistance by waving to the camera in the elevator hallway (as if he had just entered the floor).

Figure 6 shows a trajectory of waypoints visited by the robot during the execution of its computed policy, starting with the response to a waving event. In the absence of events, the robot behaved as expected, going around the floor and

visiting all the relevant rooms. However, when the robot decided to assist a visitor that was waving, it navigated to the elevator hallway where the waving was detected directly, without entering intermediate rooms.

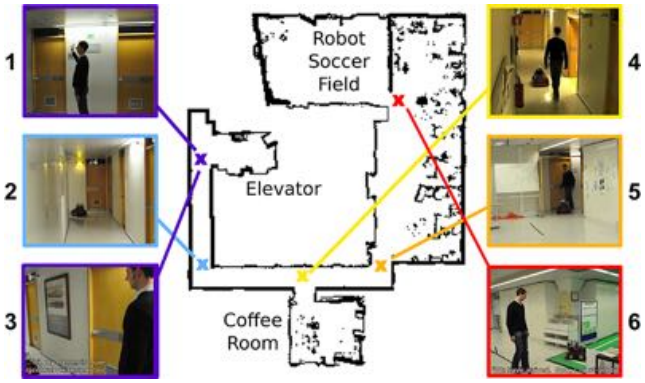

Figure 6: Assistance to visitor (with color coding the same as the topological graph described in Figure 4). When a visitor seeks assistance (waving to a camera) (1) the robot stops patrolling and goes to the event position (2) and prompts the visitor to interact (3). Once the visitor tells the robot his destination, the robot leads him there (4 and 5), notifying when the goal is reached (6).

We also simulated the MDP model to analyze the balance of the policy responding to surveillance events while patrolling. We ran the MDP for 100 steps triggering fire events uniformly at random at the Coffee Room, and repeated 500 runs for each value of triggered fires. Figure 7 shows the percentage of extinguished fires and the number of patrol rounds of the robot. The robot performs its patrol rounds and only stops them to attend and extinguish fires. As expected, as there are more fires, the robot is able to perform less rounds. Besides, some fires may be triggered close to the end of the experiment, leaving the robot with no time to reach the Coffee Room. Therefore, as the number of fires increases, the extinguishing rate gradually degrades.

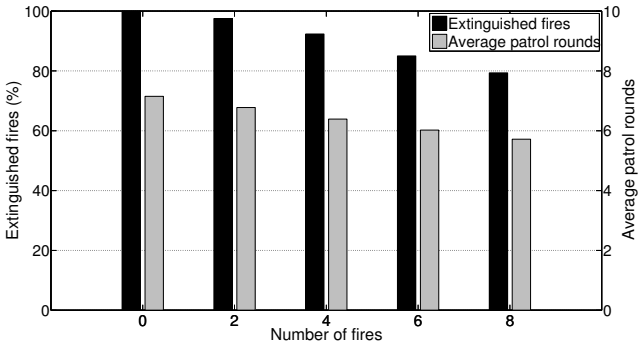

Figure 7: Testbed simulations for single-robot surveillance with increasing random fire events at the Coffee Room. Average values for the percentage of extinguished fires and the number of patrol rounds of the robot are shown.

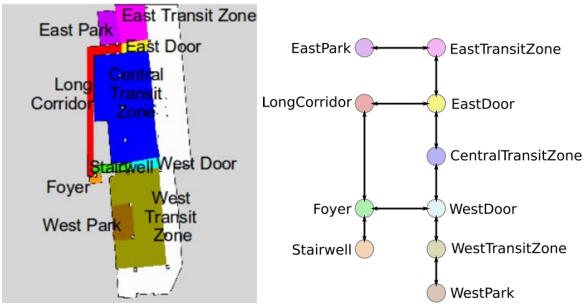

Figure 8: The topological map used at the shopping mall.

Deployment in a shopping center We performed a similar experiment in a more realistic environment located in a shopping mall. As a first step towards integration, we deployed our system in the technical corridors beneath the mall closed off to the public. The map of the scenario and its topological abstraction are shown in Figure 8. Here, in addition to waving events, trespassing events were additionally introduced. [3]

In this scenario, three functionalities of the system were tested to assess its capabilities to respond to different situations using a single balanced MDP policy. In the absence of events, the robot began moving around the environment selecting the next area to visit among those defined in Figure 8 (left), ensuring that key areas were visited frequently. During the robots' patrol, we triggered random trespassing events by entering the restricted technical corridor (see Figure 10). Each time, the robot stopped its patrol, its policy dictating that it move towards the intruder's detected position to intervene. Upon arrival, the robot requested him to "leave the area immediately". After the intruder was gone, the robot resumed its patrol. We also triggered waving events to test the robot's ability to perform visitor assist. These tests consisted of a person entering into a camera's field of view and waving with his or her hand to request help. In response to the waving detection, the robot stopped patrolling and went to the position of the event to interact with the visitors, prompting him or her to select among several possible areas in the environment. Once the visitor selected a desired destination, the robot led the way.

We carried out a third deployment of our multiagent surveillance system in the commercial, publicly accessible areas of the same shopping mall (see Figure 9). The functionalities and behaviors obtained were qualitatively identical, but the autonomous navigation of the robot was made considerably more difficult due to the characteristics of the environment and the robot's hardware limitations (for instance, glass panes of storefronts sometimes eluded its laser range finder).

Figure 9: Robot patrolling public areas of the shopping mall.

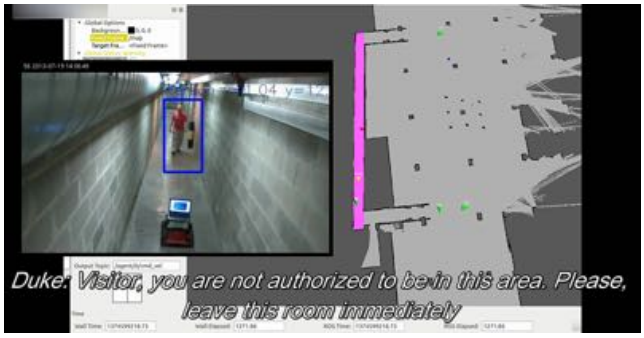

Figure 10: Interactive display showing the restricted zone and a trespassing response with the robot speaking to the intruder.

5.3 Event-Driven POMDPs for Multi-Robot Surveillance

In the next experiments, we adopt an alternative decision-making approach suitable for multi-robot settings with partial observability of event occurrences. In contrast to the MDP model, a POMDP explicitly considers that the event-detector (and hence robots' observations) are susceptible to errors. Such errors may come in the form of *false positive* detections (*e.g.* incorrectly detecting a person in an empty room) or *false negative* detections (*e.g.* failing to detect a person).

Explicitly modeling observation errors, in combination with the decisions of multiple robots, comes at a computational overhead. A conventional multi-robot POMDP is notoriously harder to solve than a regular MDP. Here, we circumvent the added complexity by considering the *hierarchical* decision-making structure shown in Figure 11. The lowest level of decision-making in our system handles the navigation of each robot to its desired poses (*i.e.* motion planning), and this is done internally by the ROS Navigation Stack. Then, a set of *tasks* defines the behaviors that each robot is capable of performing individually. Each task is not necessarily bound to a particular different decision-making formalism – in our case, we have implemented tasks

[3] A video summarizing the tests performed can be viewed at `https://youtu.be/Ivx908SSzlk` or at the multimedia attachment .

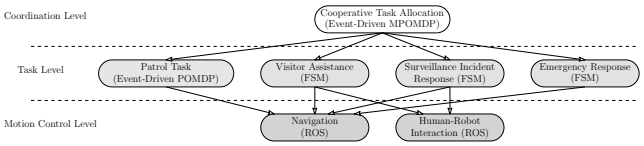

Figure 11: The various levels of decision-making involved in our multi-robot case study for autonomous surveillance.

either as manually designed Finite State Machines (FSMs), or single-robot (Event-Driven) POMDPs.

The cooperative decision-making problem in this scenario lies at the top of this hierarchical organization, and concerns the allocation of tasks between the robots, as a response to the discrete detections of the sensor network. We cast the problem of multi-robot coordination in our surveillance framework as an Event-Driven (asynchronous) Multi-robot POMDP. Multi-robot POMDPs [23] are a straightforward extension of POMDPs to multi-robot systems with free communication (which is the case in our surveillance system, since all robots share their information freely). As in an MDP, the POMDP model defines a set of states and actions; but it also defines a set of observations, which represent the possible incomplete or uncertain information that the robots have about their environment.

The actions in this multi-robot model correspond to the abstract tasks ("behaviors" in Section 5.1) that each robot must perform individually: patrol of the environment; assistance to visitors (the closest robot to the visitor should respond to the event); surveillance incident response (warning trespassers in restricted areas); and emergency response. This is the highest priority task, and should prompt robots to move to the position of the detected emergency. As with the single-robot MDP, the state space is factored into multiple variables, this time with separate variables for the local state of each robot, whether or not it is powered on, and whether or not it is busy performing a particular task (other than patrolling). As before, the rewards for each state correspond to the relative priorities of each of the three respective active events. Finally, the observations of our Multi-robot POMDP include the detection of events themselves. There is also a set of robot-specific observations (also mapped from events) that are communicated between robots to inform each other of their own local state (see the supplementary material for more details on the models).

In Figure 12, we show the timeline of a trial execution of our Event-Driven Multi-robot POMDP policy. That policy was computed for the same testbed scenario described in Figure 4 but using two Pioneer 3-AT robots. In the trial, the detection of a trespasser in a restricted area prompted one robot to inspect that position, by taking the action "Surveillance Incident Response" at step 1. Meanwhile, the other robot continued to patrol the environment; in step 2, an assistance request was detected. Since one of the robots was already busy taking care of the trespasser, the remaining robot (robot 1) decided to assist the visitor. Afterwards, the robot went back to patrolling the environment until, at step 4, a fire detection was simulated, which caused both robots to aban-

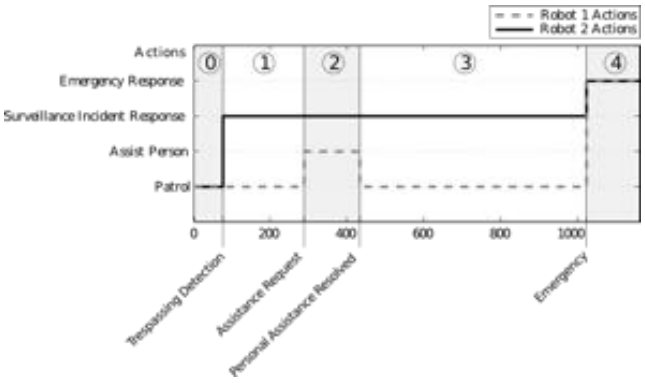

Figure 12: A timeline of actions and events in a trial run of the multi-robot case study for autonomous surveillance.

don their active tasks and address the emergency immediately. The total runtime of this trial ($19m$ $18s$) was limited only by the battery lifetimes of both robots.

Figure 13 depicts simulation results to assess our Event-Driven Multi-robot POMDP policy for the assistance of visitors. We performed experiments of fixed time length (4 hours each) while increasing the probability of false negative detections, i.e., failing to detect visitor assistance requests. Then, we measured the rate of successful visitor assistance episodes and the waiting times for those, for both the Event-Driven POMDP as well as for a baseline MDP (that assumes full observability). The results show that, as the probability of false negatives increases (and therefore the reliability of the camera network decreases), the POMDP policy is able to successfully respond to more assistance requests than the MDP baseline, since the former explicitly considers observations as stochastic, and reasons over the possibility that an undetected person is waiting for assistance. Even with complete unobservability (i.e. without ever being able to observe a request for assistance through the camera network), the POMDP policy still drives the robot periodically to check for any possible visitors. The waiting times for assisted visitors (Figure 13, bottom) are also shown to be relatively independent of the reliability of the sensors, as there is not a statistically significant difference between the respective distributions. This means that the POMDP policy induces an efficient patrol strategy that minimizes the risk that a visitor is left waiting for too long.

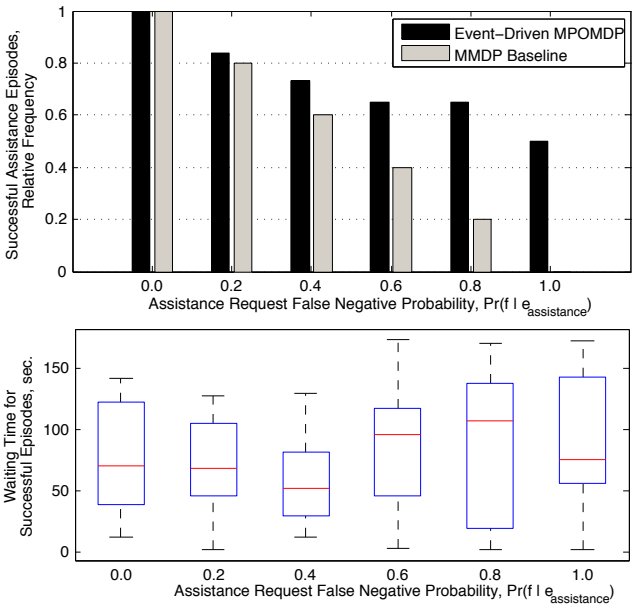

Figure 13: Testbed simulations for multi-robot surveillance increasing the probability of false negative detections of assistance requests (4 hours for each simulation). Top, average values of the rate of successful assistance episodes; bottom, boxplot of the visitor waiting times.

5.4 Limitations and Extensibility

The prototype deployments documented in the preceding sections provide proof of concept upon which future studies can build and extend beyond the system's present limitations. These limitations include, for instance, the number of robots, the richness of scenarios, and the scope of the deployment. These are not indicative of shortcomings of the surveillance framework itself, but are rather due to limited resources over the relatively short term that this project was carried out. Given substantial supplemental support, as well as the necessary permissions, a natural next step would be to operate the surveillance robots in public areas of the shopping center, leading to a more comprehensive evaluation of the performance of the system as a whole.

One might also consider limitations imposed by the robots' decision-theoretic planing methods. For instance, (PO)MDPs have the reputation of being hard to scale. Fortunately, we can mitigate the computational increase, commonly associated with adding more robots or surveilling larger areas, by employing recent research advances such as factored models [11, 21], decoupling [28], and hierarchical planning [1, 17]. More advanced methods following these paradigms are well accommodated by the surveillance framework, which already has the capacity to decentralize the robots' planning and awareness and to represent surveillance tasks with varying degrees of abstraction. In particular, note that we exploited in our case studies both factored and hierarchical models (see the supplementary material for more details).

Another challenge, that could be perceived as a limitation of the current methods used to make robot surveillance decisions, is the specification of effective MDP parameters (i.e., state feature, transition probabilities, and rewards). Such models are general enough to induce the complex behavioral policies that we have demonstrated and a wide variety of other robot behaviors. However, prescribing accurate probabilities is easier said than done in a real surveillance environment outside of the lab, where we have the limited ability to collect data with the real robots. This has since led us to consider more sophisticated modeling techniques that employ statistical inference on easy-to-collect parameters to help derive reasonable settings for hard-to-collect

parameters [29]. Similarly, we have found it nontrivial to select rewards that adequately balance competing surveillance objectives. Though preliminary advances have been made, these issues warrant further research.

6 Conclusions

The framework that we have developed constitutes an important step towards fully-autonomous surveillance. We introduce into the conventional surveillance system mobile robots that have the potential to alleviate the tasks of human operators. Our robots embody intelligent surveillance nodes capable of pursing a variety of surveillance activities and of deciding among activities in real time based on the occurrence and urgency of events in a dynamic and uncertain environment. Underlying the robots' autonomy is a framework architecture that automatically detects anomalies, aggregates and filters detections to interpret them as events, transmits those events to the robot, and responds by intelligent reasoning, navigation, and physical interaction.

This is all made possible by leveraging several complementary research technologies such as computer vision, robot automation, and intelligent decision making, and integrating them into a cohesive, modular design. Our case studies demonstrate a progression towards increasingly complex scenarios in increasingly realistic surveillance environments, whereby we have been able to take our system out of the lab and into a shopping center.

However, the primary benefit of our framework is that it serves as a research platform with which to apply decision-making formalisms and techniques to a real robot problem. Autonomous surveillance is a rich domain wherein resource constraints, and uncertainties, and competing objectives, provide significant challenges that can be addressed through decision-theoretic planning. This has driven us to develop solutions using MDPs and POMDPs as described in our case studies, pushing the state of art and developing novel advances for planning in real world settings [17, 18, 29].

Acknowledgements

This work was partially supported by the Portuguese Fundação para a Ciência e a Tecnologia (FCT), through strategic funding for ISR/LARSyS under project PEst-OE/EEI/LA0021/2013 and through the Carnegie Mellon-Portugal Program under project CMU-PT/SIA/0023/2009. This work was also partially supported by national funds through Fundação para a Ciência e a Tecnologia (FCT) with reference UID/CEC/50021/2013. This work was also partially funded by project FCT UID/EEA/50009/2013 of ISR/LARSyS.

Author Information

Stefan Witwicki, Robotic Systems Laboratory, Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland. E-mail: witwicki@gmail.com.

José Carlos Castillo, Department of Systems Engineering and Automation, University Carlos III of Madrid, Spain. E-mail: jocastil@ing.uc3m.es.

João Messias, Intelligent Systems Lab, University of Amsterdam, The Netherlands. E-mail: jmessias@uva.nl.

Jesús Capitán, Robotics, Vision and Control Group, University of Seville, Spain. E-mail: jcapitan@us.es.

Francisco S. Melo, INESC-ID, and Instituto Superior Técnico (IST), University of Lisbon, Portugal. E-mail: fmelo@inesc-id.pt.

Pedro U. Lima, Institute for Systems and Robotics (ISR), Instituto Superior Técnico (IST), University of Lisbon, Portugal. E-mail: pal@isr.tecnico.ulisboa.pt.

Manuela Veloso, School of Computer Science, Carnegie Mellon University (CMU), Pittsburgh, USA. E-mail: mmv@cs.cmu.edu.

References

[1] Amato, C.; Konidaris, G. D.; and Kaelbling, L. P. 2014. Planning with macro-actions in decentralized POMDPs. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, 1273–1280. International Foundation for Autonomous Agents and Multiagent Systems.

[2] Barbosa, M.; Bernardino, A.; Figueira, D.; Gaspar, J.; Goncalves, N.; Lima, P.; Moreno, P.; Pahliani, A.; Santos-Victor, J.; Spaan, M.; and Sequeira, J. 2009. Isrobotnet: A testbed for sensor and robot network systems. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, 2827–2833.

[3] Bellman, R. 1957. *Dynamic Programming*. Princeton University Press.

[4] Chen, P.; Ahammad, P.; Boyer, C.; Huang, S.-I.; Lin, L.; Lobaton, E.; Meingast, M.; Oh, S.; Wang, S.; Yan, P.; Yang, A.; Yeo, C.; Chang, L.-C.; Tygar, J.; and Sastry, S. 2008. Citric: A low-bandwidth wireless camera network platform. In *Distributed Smart Cameras, 2008. ICDSC 2008. Second ACM/IEEE International Conference on*, 1–10.

[5] Cucchiara, R.; Grana, C.; Prati, A.; Tardini, G.; and Vezzani, R. 2004. Using computer vision techniques for dangerous situation detection in domotic applications. In *Intelligent Distributed Surveilliance Systems, IEE*, 1–5.

[6] del Corte, A.; Gutierrez, O.; and Gomez, J. 2012. New location techniques based on ray-tracing for increasing airport safety in apron and stand areas. *Frontiers in Computer Education: Advances in Intelligent and Soft Computing* 133:515–522.

[7] Delle Fave, F.; Canu, S.; Iocchi, L.; Nardi, D.; and Ziparo, V. 2009. Multi-objective multi-robot surveillance. In *Autonomous Robots and Agents, 2009. ICARA 2009. 4th International Conference on*, 68–73.

[8] Fernández-Caballero, A.; Castillo, J. C.; López, M. T.; Serrano-Cuerda, J.; and Sokolova, M. V. 2013. Int3-horus framework for multispectrum activity interpretation in intelligent environments. *Expert Systems with Applications* 40(17):6715–6727.

[9] Galindo, C.; Fernández-Madrigal, J.-A.; González, J.; and Saffiotti, A. 2008. Robot task planning using semantic maps. *Robotics and Autonomous Systems* 56(11):955–966.

[10] Gerkey, B. P., and Matarić, M. J. 2004. A formal analysis and taxonomy of task allocation in multi-robot systems. *The International Journal of Robotics Research* 23(9):939–954.

[11] Guestrin, C.; Koller, D.; and Parr, R. 2001. Multiagent planning with factored mdps. In *NIPS*, volume 1, 1523–1530.

[12] Ke, Y.; Sukthankar, R.; and Hebert, M. 2005. Efficient visual event detection using volumetric features. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, 166–173. IEEE.

[13] Ke, Y.; Sukthankar, R.; and Hebert, M. 2007. Spatio-temporal shape and flow correlation for action recognition. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, 1–8. IEEE.

[14] Kullback, S., and Leibler, R. 1951. On information and sufficiency. *Annals of Mathematical Statistics* 22(1):79—-86.

[15] Lemaire, T.; Alami, R.; and Lacroix, S. 2004. A distributed tasks allocation scheme in multi-uav context. In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, volume 4, 3622–3627. IEEE.

[16] Maheswaran, R. T.; Tambe, M.; Bowring, E.; Pearce, J. P.; and Varakantham, P. 2004. Taking dcop to the real world: Efficient complete solutions for distributed multi-event scheduling. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 1*, 310–317. IEEE Computer Society.

[17] Messias, J.; Spaan, M.; and Lima, P. 2013. GSMDPs for multi-robot sequential decision-making. In *AAAI Conference on Artificial Intelligence*, 1408–1414.

[18] Messias, J. 2014. *Decision-Making under Uncertainty for Real Robot Teams*. Ph.D. Dissertation, Instituto Superior Técnico.

[19] Moreno, P.; Bernardino, A.; and Santos-Victor, J. 2009. Waving detection using the local temporal consistency of flow-based features for real-time applications. In *Image Analysis and Recognition*. Springer. 886–895.

[20] Niebles, J. C.; Wang, H.; and Fei-Fei, L. 2008. Unsupervised learning of human action categories using spatial-temporal words. *International journal of computer vision* 79(3):299–318.

[21] Oliehoek, F. A.; Whiteson, S.; and Spaan, M. T. J. 2013. Approximate solutions for factored Dec-POMDPs with many agents. In *AAMAS13*, 563–570.

[22] Onut, V.; Aldridge, D.; Mindel, M.; and Perelgut, S. 2010. Smart surveillance system applications. In *Proceedings of the 2010 Conference of the Center for Advanced Studies on Collaborative Research*, 430–432.

[23] Pynadath, D. V., and Tambe, M. 2002. The communicative multiagent team decision problem: Analyzing teamwork theories and models. *Journal of Artificial Intelligence Research* 16(1):389–423.

[24] Quigley, M.; Conley, K.; Gerkey, B.; Faust, J.; Foote, T.; Leibs, J.; Wheeler, R.; and Ng, A. Y. 2009. ROS: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, 5.

[25] Schüldt, C.; Laptev, I.; and Caputo, B. 2004. Recognizing human actions: a local svm approach. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 3, 32–36. IEEE.

[26] Smallwood, R. D., and Sondik, E. J. 1973. The optimal control of partially observable markov processes over a finite horizon. *Operations Research* 21(5):1071–1088.

[27] Svenonius, O. 2012. The stockholm security project: Plural policing, security and surveillance. *Information Polity* 17(1):35–43.

[28] Witwicki, S. J., and Durfee, E. H. 2011. Towards a unifying characterization for quantifying weak coupling in Dec-POMDPs. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, 29–36.

[29] Witwicki, S. J.; Melo, F. S.; Capitán, J.; and Spaan, M. T. 2013. A flexible approach to modeling unpredictable events in MDPs. In *Proceedings of the 23rd International Conference on Automated Planning and Scheduling (ICAPS-2013)*, 260–268.

[30] Yao, Y.; Chen, C.-H.; Koschan, A.; and Abidi, M. 2010. Adaptive online camera coordination for multi-camera multi-target surveillance. *Computer Vision and Image Understanding* 114(4):463–474.

Supplementary Material for Autonomous Surveillance Robots
A Decision-Making Framework for Networked Muiltiagent Systems

S. Witwicki, J.C. Castillo, J. Messias, J. Capitán, F. Melo, P.U. Lima and **M. Veloso**

A Supplementary Material

In this appendix, we present auxiliary material regarding the definition and models of the decision-theoretic planners used for the case studies of the paper.

A.1 MDPs for Single-robot Surveillance

In this section, we describe the decision-making models that were used in our single-robot surveillance case study (Section 5.2). This case study was modeled as a factored MDP with the 2-stage Dynamic Bayesian Network (DBN) depicted in Figure 1. The factored state space description and the action space are represented in Figure 2. We show here the MDP for the deployment in the testbed, whose topological map is shown in Figure 4. There, we considered two types of events: a visitor requesting assistance and an emergency. More details about this MDP model for single-robot surveillance can be found in [2]. Moreover, the MDP for the deployment in the shopping center would be analogous, but based on the topological map of Figure 8 and considering also the event of someone trespassing a restricted area (an action for expelling the intruders is also added).

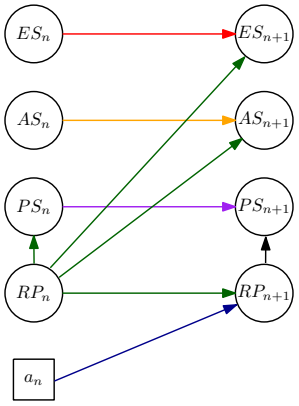

Figure 1: The 2-DBN for our single-robot MDP. Outgoing connections from the same node at time n are represented with the same color, for better visibility. The represented state factor variables are: "Robot Position" (RP); "Patrol Status" (PS); "Assistance Request Status" (AS) "Emergency Status" (ES).

$$\text{Robot Position: } \mathcal{S}_1 = \left\{ \begin{array}{l} \text{Elevator Hallway} \\ \text{East Corridor} \\ \text{Coffee Room} \\ \text{North Corridor} \\ \text{Mobile Robotics Lab} \\ \text{Soccer Field} \end{array} \right.$$

$$\text{Patrol Status: } \mathcal{S}_2 = \left\{ \begin{array}{l} 000 \\ 001 \\ 010 \\ 011 \\ 100 \\ 101 \\ 110 \\ 111 \end{array} \right.$$

$$\text{Assistance Status: } \mathcal{S}_3 = \left\{ \begin{array}{l} \text{None} \\ \text{Waiting at Elevator Hallway} \end{array} \right.$$

$$\text{Emergency Status: } \mathcal{S}_4 = \left\{ \begin{array}{l} \text{None} \\ \text{Fire at Coffee Room} \end{array} \right.$$

$$\mathcal{A} = \left\{ \begin{array}{l} \text{Up} \\ \text{Down} \\ \text{Left} \\ \text{Right} \end{array} \right.$$

Figure 2: State and action spaces description for the single-robot MDP.

There is a binary factor variable for each possible event, indicating whether there is an unattended event or not. In the example shown in this appendix, the visitor can only arrive at the Elevator Hallway, asking for assistance there; and the emergency can only take place at the Coffee Room, due to a potential fire. The events are attended once the robot visits their corresponding locations. For the patrol task, we define three hotspots that the robot has to visit at each round: the Elevator Hallway, the Coffee Room and the Soccer Field. The factor variable PS takes 8 possible values that are the binary representation of a 3-bit sequence corresponding to 3 flags indicating the hotspots already visited. When one of the hotspots is visited, the corresponding flag is set to 1; whenever $PS_n = 111$, then $PS_{n+1} = 000$, indicating that the robot should repeat its patrol round after visiting all hotspots.

The MDP has infinite horizon with a discount factor 0.95; we reward states where the robot has visited all the hotspots, and penalize those where an event is unattended. By weighting those rewards properly, we can give more priority to certain events and leave the robot patrolling when there are no events. The patrol order is not explicitly enforced in the model, but movement actions have a cost, leading the robot to take the shortest path. Moreover, the exact values for the rewards were chosen empirically, by systematically varying the weights, observing the behavior of the resulting MDP policies, and selecting those values that achieved a balance between patrolling and attending to the various events.

A.2 Event-Driven POMDPs for Multi-Robot Surveillance

In this section, we describe the decision-making models that were used in our multi-robot surveillance case study (Section 5.3). There, a hierarchical approach was proposed with a top-level event-driven MPOMDP and several task-level planners. We describe here the states, actions and observations for the event-driven POMDPs of the top-level and the patrol task. Due to their size, we omit the descriptions of the FSMs, but all details can be found in [1].

Event-driven POMDPs, such as those used to model our cooperative surveillance task, are especially suited to model high-level decision-making tasks, in which decisions do not need to be taken at a fixed rate, but rather at random instants separated by arbitrarily long time intervals, and marked by the occurrence of an appropriately defined "event". When conditioned on the occurrence of each possible event, the transitions of event-driven POMDPs are usually (but not necessarily) sparse and deterministic. Therefore, the modeling of these event-driven POMDPs mostly depends on estimating the (time-independent) probability of occurrence of each event. For the purpose of our demonstrative surveillance task, which involves events that have to be simulated in practice due to their nature (such as the occurrence of emergencies or the appearance of trespassers in our laboratory) these events were given fixed probabilities *a priori*. We emphasize, however, that in a real-world task these values could be estimated from data on the actual occurrence of such events.

Observations in an event-driven model, in turn, correspond to the ability to correctly detect that an event has occurred. The probabilities of correct and incorrect observations (false positives and negatives) were empirically estimated based on the multi-agent system to observe each of the possible events. In our system, we used infinite horizon policies with a discount factor of 0.95.

Coordinative (top-level) event-driven MPOMDP The graphical model for the Coordinative Event-Driven MPOMDP that was used to allocate tasks in our multi-robot team is shown in Figure 3. The presence of a decoupling "Event prior" variable is a characteristic feature of an event-driven model. Since, in such models, variables typically change their values asynchronously (rather than simultaneously at fixed time instants), the use of a decoupling variable at time $t+1$ allows the conditional dependencies between all other variables in the model to become sparse, which otherwise would not be possible.

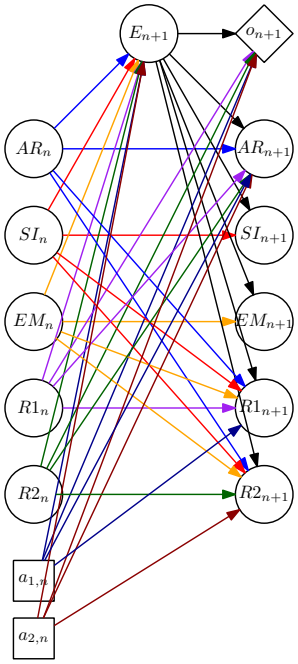

Figure 3: The 2-DBN for our Coordinative Event-Driven MPOMDP, which assigns tasks to each robot. Outgoing connections from the same node at time n are represented with the same color, for better visibility. The represented state factor variables are: "Assistance Requested" (AR'); "Surveillance Incident" (SI); "Emergency" (EM); "Robot 1 Status" (R1); "Robot 2 Status" (R2); "Event Prior" (E).

The action and observation spaces of our top-level model are described in Figure 5. In this model, the reward structure assigned different costs (negative rewards) to the occurrence of emergencies, surveillance incidents, and assistance requests, to capture their relative priorities. States with none of these occurrences were given 0 reward (the maximal reward for the problem). Therefore, the induced behavior is to resolve each of the ongoing occurrences according to their relative value.

Patrol task event-driven POMDP The "Patrol" task Event-Driven POMDP was implemented as the graphical model shown in Figure 6. In Figure 7 we show the factored state description for this problem, and in Figure 8 we describe its action and observation spaces.

$$\text{Assistance Requested: } \mathcal{S}_1 = \begin{cases} \text{No} \\ \text{Robot 1 Assisting} \\ \text{Robot 2 Assisting} \\ \text{Yes} \end{cases}$$

$$\text{Surveillance Incident: } \mathcal{S}_2 = \begin{cases} \text{No} \\ \text{Yes} \end{cases}$$

$$\text{Emergency: } \mathcal{S}_3 = \begin{cases} \text{No} \\ \text{Yes} \end{cases}$$

$$\text{Robot 1 Status: } \mathcal{S}_4 = \begin{cases} \text{Disabled} \\ \text{Idle} \\ \text{Busy} \end{cases}$$

$$\text{Robot 2 Status: } \mathcal{S}_5 = \begin{cases} \text{Disabled} \\ \text{Idle} \\ \text{Busy} \end{cases}$$

Figure 4: State space description for our Coordinative Event-Driven MPOMDP.

$$\mathcal{A}_1 = \mathcal{A}_2 = \begin{cases} \text{Patrol} \\ \text{Assist Person} \\ \text{Surveillance Incident Resp.} \\ \text{Emergency Response} \end{cases}$$

$$\mathcal{O} = \begin{cases} \text{False Negative} \\ \text{Timeout} \\ \text{Waving - Low Confidence} \\ \text{Waving - High Confidence} \\ \text{Surveillance Incident} \\ \text{Emergency} \\ \text{Person Assistance Resolved} \\ \text{Surveillance Incident Resolved} \\ \text{Emergency Resolved} \\ \text{Robot 1 Task Cancelled} \\ \text{Robot 2 Task Cancelled} \\ \text{Robot 1 On / Off} \\ \text{Robot 2 On / Off} \end{cases}$$

Figure 5: Action space description, top, and observation space description, bottom, for the Coordinative Event-Driven MPOMDP.

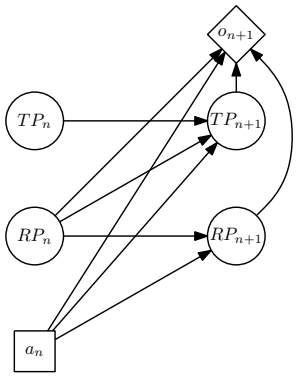

Figure 6: The 2-DBN for the "Patrol" Event-Driven POMDP. The represented state factor variables are: "Robot Position" (RP); "Target Position" (TP).

$$\text{Robot Position: } \mathcal{S}_1 = \begin{cases} \text{Elevator Hallway} \\ \text{East Corridor} \\ \text{Coffee Room} \\ \text{North Corridor} \\ \text{Mobile Robotics Lab} \\ \text{Soccer Field} \end{cases}$$

$$\text{Target Position: } \mathcal{S}_2 = \begin{cases} \text{No Target} \\ \text{Elevator Hallway} \\ \text{East Corridor} \\ \text{Coffee Room} \\ \text{North Corridor} \\ \text{Mobile Robotics Lab} \\ \text{Soccer Field} \end{cases}$$

Figure 7: State space description for our "Patrol" task Event-Driven POMDP. See Figure 4 for the semantic grounding of these labels.

$$\mathcal{A} = \begin{cases} \text{Up} \\ \text{Down} \\ \text{Left} \\ \text{Right} \\ \text{Expel Intruder} \\ \text{Report Area Clear} \end{cases}$$

$$\mathcal{O} = \begin{cases} \text{Elevator Hallway Clear} \\ \text{East Corridor Clear} \\ \text{Coffee Room Clear} \\ \text{North Corridor Clear} \\ \text{Mobile Robotics Lab Clear} \\ \text{Soccer Field Clear} \\ \text{Found Target} \end{cases}$$

Figure 8: Action space description, top, and observation space description, bottom, for the "Patrol" task Event-Driven POMDP.

Author Information

Stefan Witwicki, Robotic Systems Laboratory, Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland. E-mail: stefan.witwicki@epfl.ch.

José Carlos Castillo, Department of Systems Engineering and Automation, University Carlos III of Madrid, Spain. E-mail: jocastil@ing.uc3m.es.

João Messias, Intelligent Systems Lab, University of Amsterdam, The Netherlands. E-mail: jmessias@uva.nl.

Jesús Capitán, Robotics, Vision and Control Group, University of Seville, Spain. E-mail: jcapitan@us.es.

Francisco S. Melo, INESC-ID, and Instituto Superior Técnico (IST), University of Lisbon, Portugal. E-mail: fmelo@inesc-id.pt.

Pedro U. Lima, Institute for Systems and Robotics (ISR), Instituto Superior Técnico (IST), University of Lisbon, Portugal. E-mail: pal@isr.tecnico.ulisboa.pt.
Manuela Veloso, School of Computer Science, Carnegie Mellon University (CMU), Pittsburgh, USA. E-mail: mmv@cs.cmu.edu.

References

[1] Messias, J. 2014. *Decision-Making under Uncertainty for Real Robot Teams*. Ph.D. Dissertation, Instituto Superior Técnico.

[2] Witwicki, S. J.; Melo, F. S.; Capitán, J.; and Spaan, M. T. 2013. A flexible approach to modeling unpredictable events in MDPs. In *Proceedings of the 23rd International Conference on Automated Planning and Scheduling (ICAPS-2013)*, 260–268.