This is a postprint version of the following published document:


DOI: 10.1109/mra.2017.2662222

© 2017, IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.
**Autonomous Surveillance Robots**

**A Decision-Making Framework for Networked Multiagent Systems**


---

**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 transport networks \[27\] and public spaces \[3\], 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 \[24\]. 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 \[5\]. 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 fulfill a combination of objectives given limited resources. This is often framed as a (multi-)robot task selection (and allocation) problem \[19\], and has been approached through a variety of AI techniques: from logic-based (classical) planning methods \[7\], to market (auction)-based solutions \[15\] and those based on constraint optimization \[13\]. 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 unpredictable events detected over fixed camera streams, thereby leveraging real-time scene analysis.
A trespassing event is 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.

### 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.

#### Table 1: Challenges of surveillance decision-making

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

Figure 1: A staged indoor surveillance environment with the positions of the static cameras (red circles) and the common coordinate system for event location.
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 movement 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 [13].

<table>
<thead>
<tr>
<th>Technique</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our method</td>
<td>91.7%</td>
</tr>
<tr>
<td>Ke et al. [13]</td>
<td>91.7%</td>
</tr>
<tr>
<td>Ke et al. [12]</td>
<td>91.7%</td>
</tr>
<tr>
<td>Schuldt et al. [14]</td>
<td>73.6%</td>
</tr>
</tbody>
</table>

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

(Continued with our running example, Figure 1(a) highlights two cameras located in the area labeled as north corridor with overlapping fields of view. Figure 1(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 1). 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.
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 3. 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 expressively 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 $s_t \in S$ 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” $a_t \in A_t$, where $A_t$ 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 in $s_t = \langle r_1, x_1, \ldots, x_n, b_1 \rangle$, where $r_i$ represents the topological position of the robot (within the possible alternatives represented in Figure 3), $x_1, \ldots, x_n$ are the states of each topological node, which could be modeled, for instance, as...
The optimization problem at each time step, given the states, the general decision-making process can be cast as the following optimization problem: at each time \( t \), given the history of states and actions \( (s_0, a_0, s_1, a_1, \ldots, s_t, a_t) \), select a new action \( a_{t+1} \) 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, \( \pi : S \to A \) (a policy) that maximizes the expected accumulated reward over a possibly infinite number of future steps (i.e., utility-based planning).

The definition of the components of an MDP is domain-independent. 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 [29], 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 5.3 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 4 on top of the widely-adopted ROS infrastructure [29]. Our implementation consists of three macro-blocks communicating by means of ROS topics (see Figure 4). 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 5 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.](image)

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 [1]). 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 un-der 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.

**Assistance to visitors.** As noted in Section [2], 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] 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 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 [1]. 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 [2] 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

[^2]: The MDM package: [http://wiki.ros.org/markov_decision_making](http://wiki.ros.org/markov_decision_making)
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.

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.

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).
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-detection (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 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 9 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.

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

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.
parameters. 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.

Acknowledgements

This work was partially supported by the Portuguese Fundação para a Ciência e a Tecnologia (FCT), through Paloha, through the Carnegie Mellon-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 of ISR/LARSyS.

Author Information

Stefan Witwicki. Robotic Systems Laboratory, Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland. E-mail: witwicki8@gmail.com.

José Carlos Castro. Department of Systems Engineering and Automation, University Carlos III of Madrid, Spain. E-mail: jccastil@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: fme@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: mvel@cs.cmu.edu.

References


