Chapter 8
Multi-robot Teams for Environmental Monitoring

Maria Valera Espina, Raphael Grech, Deon De Jager, Paolo Remagnino, Luca Iocchi, Luca Marchetti, Daniele Nardi, Dorothy Monekosso, Mircea Nicolescu, and Christopher King

Abstract. In this chapter we target the problem of monitoring an environment with a team of mobile robots having on board video-cameras and fixed stereo cameras available within the environment. Current research regards homogeneous robots, whereas in this chapter we study highly heterogeneous systems and consider the problem of patrolling an area with a dynamic set of agents. The system presented in the chapter provides enhanced multi-robot coordination and vision-based activity monitoring techniques. The main objective is the integration and development of coordination techniques for multi-robot environment coverage, with the goal of maximizing the quality of information gathered from a given area thus, implementing a *Heterogeneous mobile and reconfigurable multi-camera video-surveillance system*.

8.1 Introduction

Monitoring a large area is a challenging task for an autonomous system. During recent years, there has been increasing attention in using robotic technologies for security and defense applications, in order to enhance their performance and reduce the danger for the people involved. Moreover, the use of multi-robot systems allows for a better deployment, increased flexibility and reduced costs of the system. A significant amount of research in multi-agent systems have been dedicated to the development of and experimentation on methods, algorithms and evaluation methodologies for multi-robot patrolling in different scenarios. This chapter shows

Maria Valera Espina · Raphael Grech · Deon De Jager · Paolo Remagnino
Digital Imaging Research Centre, Kingston University, London, UK

Luca Iocchi · Luca Marchetti · Daniele Nardi
Department of Computer and System Sciences University of Rome "La Sapienza", Italy

Dorothy Monekosso
Computer Science Research Institute, University of Ulster, UK

Mircea Nicolescu · Christopher King
Department of Computer Science and Engineering, University of Nevada, Reno

P. Remagnino et al. (Eds.): Innovations in Defence Support Systems – 3, SCI 336, pp. 183–209.
springerlink.com

that using multi-robots in environmental monitoring is both effective and efficient. We provide a distributed, multi-robot solution to environment monitoring, in order to detect or prevent defined, undesired events, such as intrusions, leaving unattended luggage and high temperatures (such as a fire). The problem of detecting and responding to threats through surveillance techniques is particularly well suited to a robotic solution comprising of a team of multiple robots. For large environments, the distributed nature of the multi-robot team provides robustness and increased performance of the surveillance system. Here we develop and test an integrated multi-robot system as a *mobile, reconfigurable, multi-camera video-surveillance system*. The system goal is to monitor an environment by collectively executing the most effective strategies for gathering the best quality information from it. Using a group of mobile robots equipped with cameras has several significant advantages over a fixed surveillance camera system. Firstly, our solution can be used in environments that have previously not been equipped with a camera-based monitoring system: the robot team can be deployed quickly to obtain information about an unknown environment. Secondly, the cameras are attached to the robots, which will be positioning themselves within the environment, in order to best acquire the necessary information. This is in contrast with a static camera, which can only perform observations from a fixed view point. Thirdly, the robots in the team have the power to collaborate on the monitoring task and are able to pre-empt a potential threat. Fourthly, the robots could be equipped with additional, specialized sensors, which could be delivered at the appropriate place in the environment to detect, for example, the presence of high temperatures, such as in the case of a fire. Lastly, the robot team can communicate with a human operator and receive commands about the goals and potential changes in the mission, allowing for a dynamic, adaptive solution. Therefore, these enhanced multi-robot coordination and vision-based activity monitoring techniques, advance the state-of-the-art in surveillance applications. In this chapter, we focus on monitoring a large area by using a system with the following characteristics.

1. The system is composed of a number of agents, some of them having mobile capabilities (mobile robots) whilst others are fixed (video cameras).
2. The system is required to monitor and detect different kinds of predefined events at the same time.
3. Each agent has a set of sensors that are useful to detect some events. Sensors are of a different type within the entire system.
4. The system is required to operate in two modes:
   a) patrolling mode
   b) response mode

These requirements make the problem significantly different from previous work. First of all, we consider a highly heterogeneous system, where robots and cameras inter-operate. Second, we consider different events and different sensors and we will therefore consider different sensor models for each kind of event. Third, we will study the dynamic evolution of the monitoring problem, where at each time a subset of the agents will be in response mode, while the rest of them will be in patrolling mode.

The main objectives of the developed system are:

1. develop environment monitoring techniques through behavior analysis based on stereo cameras,
2. develop distributed multi-robot coverage techniques for security and surveillance,
3. validate our solution by constructing a technological demonstrator showing the capabilities of a multi-robot system to effectively deploy itself in the environment and monitor it.

In our previous work, we already developed and successfully implemented new dynamic distributed task assignment algorithms for teams of mobile robots: applied to robotic soccer [27] and for foraging-like tasks [20]. More specifically, in [27] we proposed a greedy algorithm to effectively solve the multi-agent dynamic and distributed task assignment problem, that is very effective in situations where the different tasks to be achieved have different priorities. In [20] we also proposed a distributed algorithm for dynamic task assignment based on token passing that is applicable when tasks are not known a priori, but are discovered during the mission. The problem considered here requires both finding an optimal allocation of tasks among the robots and taking into account tasks that are discovered at runtime. Therefore it is necessary to integrate the two approaches. As a result, we do not only specialize these solutions to the multi-robot surveillance and monitoring task, but also study and develop extensions to these techniques in order to improve the optimality of the solutions and the adaptivity to an open team of agents, taking into account the physical constraints of the environment and of the task.

The use of stereo cameras in video-surveillance opens several research issues, such as the study of segmentation algorithms based on depth information provided by the stereo-vision, tracking algorithms that take into account 3-D information about the moving objects and techniques of behavior analysis that integrate and fuse 3-D information gathered from several stereo sensors. Furthermore, the application of multi-robot coverage to security and surveillance tasks provide new opportunities of studying multi-robot distributed coordination techniques with dynamic perception of the tasks and methods for optimal coverage of the environment in order to maximize the quality of the information gathered from it.

These aspects will be considered in more detail in the coming sections. The rest of the chapter is organized as follows: Section 8.2 provides an overview of our proposed system. In Section 8.3 previous related work is presented. The Representation formalism is explained in Section 8.4 and the event-driven multi-sensor monitoring algorithm presented in Section 8.5. In Section 8.6 the system implementation and experimental results are illustrated. Finally Conclusions are drawn in Section 8.7.

8.2 Overview of the System

The overall surveillance system developed is presented hereunder. The system is mainly composed of two subsystems: a video-surveillance sub-system operating with static cameras and a multi-robot system for environment monitoring and threat response.

8.2.1 Video-Surveillance with Static Cameras

One objective of the visual surveillance system was to identify when people leave, pick or exchange objects. Two scenarios were used to test the capabilities of the video-surveillance system. In the first scenario (i.e. unattended baggage event), the system was designed to send a report if a person was observed leaving a bag In the second scenario (i.e. object manipulation), the system should send a report if a person manipulated an unauthorized object from the environment. Once a report is sent, a patrol robot would be commissioned to go and take a high resolution picture of the scenario. Recognizing these types of actions may be done without sophisticated algorithms, so for this demonstration, we use a simple rule-sets based only on proximity and trajectories: For the first scenario:

- If a bag appears in the environment, models will be generated for that bag and for the nearest person. If the associated person is observed moving away from the bag, it will be considered “left bag”, and a report of the incident will be generated.
- If a bag is associated with one person, and a second person is observed moving away with the bag, it will be considered an “bag taken”and a report will be generated and sent to the multi-robot system.

For the second scenario:

- If a person is observed manipulating an object that was either present at the beginning of the sequence, or left by another person (i.e. unauthorized object), the incident will be considered an “allert”and a report will be generated and sent to the multi-robot system.

8.2.2 Multi-robot Monitoring of the Environment

Our approach to multi-robot monitoring is to extend the work done in multi-robot patrolling, adding the capability for the robots to respond to events detected by visual and other sensors in a coordinated way. Therefore two problems are considered and solved.

1. Identify global tasks associated to events detected by local sensors on-board the robots or the vision components of the system.
2. Coordinate the response to these events among the multiple robots.

These problems have been solved by developing a general algorithm for event-driven distributed monitoring (see Section 8.5).

8.2.3 Experimental Scenario

The scenario used for the experimental validation was tested in the campus of the Department of Computer and System Science (DIS) of Sapienza University in Rome, Italy[1]. The selected scenario, shown in Figure 8.1, was an indoor corridor to

[1] www.dis.uniroma1.it

simulate the unattended baggage event and a lab room to simulate the object manipulation. A team of robots carrying video cameras is deployed in the environment as they cooperate to optimize the surveillance task, by maximizing the amount and quality of information gathered from the environment using the on-board cameras. When the robots reach the desired target poses, the cameras mounted on them could act as a network of surveillance cameras and video-surveillance algorithms may run on their inputs. Moreover, another system based on fixed stereo cameras, capable of providing depth information, is available within the environment. This can eventually be also integrated on the robot platforms.

Fig. 8.1 Experimental scenario at DIS

8.3 Related Work

In this chapter, we define the problem of Environmental Monitoring by extending the classical problem of Multi-Robot Patrolling to include also the Threat Response, i.e. the response to a threat event detected within the environment by an agent (either a robot or a computer vision sub-system) monitoring that environment. Examples of threat responses can vary from intercepting an intruder, examining a specific area or a specific object left by somebody. The main components to be integrated for the effective monitoring of an environment are: Multi-Robot Patrolling, Multi-Robot Coverage, Dynamic Task Assignment and Automatic Video Surveillance. The current state-of-the-art about these topics is presented in the following sections.

8.3.1 Multi-robot Patrolling

The *patrolling* task refers to the act of walking around an area, with some regularity, in order to protect or supervise it [39]. Usually, it is carried on by a team of multiple robots to increase the efficiency of the system.

Given a map representation of the environment, the first step is to define whether the map should be partitioned, or not, in smaller sections. In order to maximize the efficiency, a Multi-Robot team should assign to each robot different areas to be patrolled. That means that a division of the global map has to be done, and the sub-maps assigned to the robots. As analyzed in [39], in most cases it is sufficient to adopt a static strategy, wherein the whole environment is already given as a collection of smaller areas. This means that a partitioning is not really necessary. However, more interesting approaches deal with dynamic and non-deterministic environment, resulting in a more challenging domain, that requires to be partitioned dynamically. This fact involves that the robots should coordinate themselves, to decide who has to patrol which area. The subsequent step involves how to sweep the assigned area. Basically, this is performed by transforming the environment map into a graph to be traversed. This aspect of patrolling is the most coped by the current state-of-the-art. In fact, given a topological graph the map, most of the algorithms and techniques used for dealing with graph can be adopted. Major approaches use the Traveling Salesman Problem and its variant to define an optimal (or sub-optimal) path for a given graph. In [38] is defined a steering control based mechanism that takes into account the constraint given by a real platform to define the path. First, a rectangle partitioning schema is applied to the map. Then, each rectangle is covered by the circle defining the sweep area (it depends on platform and sensors used). Finally, the path is the result of connecting the covering circles. Another possibility is given by using Hamiltonian cycle to define the path. The work in [37] defines an algorithm to transform an occupancy grid based map into a topological graph, where this kind of partitioning strategies are applied to perform the patrol task. More advanced techniques apply Game Theory [6] and Reinforcement Learning [4] methods, to include the behavior of intruder in the sequencing strategies. A comparison of this techniques and preliminary results are presented in [4]. The last aspect involved in Multi-Robot Patrolling is the task reallocation. When dynamic domain are utilized as test bed, the assigned area can change over time. This fact implies that the patrolling team needs to reshape the strategy to take into account the modification. Usually, it involves rebuilding the topological graph, and resetting the current configuration. A more efficient approach, however, requires a coordination among the robots, to minimize the task hopping. A basic approach, involving reallocation over team formation, is presented in [1].

8.3.2 Multi-robot Coverage

The goal of the *coverage* task is to build an efficient path to ensure the whole area is crossed by the robot. Using a team of robots, the goal requires to build efficient paths to jointly ensure the coverage of the area. Therefore, an important issue in mobile Multi-Robot teams is the application of coordinating tasks to the area coverage problem.

Multi-Robot environment coverage has been recently studied by solving the problem of generating patrol paths. Elmaliach et al. [18] introduced an algorithm that guarantees the maximal uniform frequency for visiting places by the robots. Their algorithm detects circular paths that visit all points in the area, while taking into account terrain directionality and velocity constraints. The approach in [2] considers also the case in which the adversary knows the patrol scheme of the robots. Correll and Martinoli [17] consider the combination of probabilistic and deterministic algorithms for the multi-robot coverage problem. They apply their method to a swarm-robotic inspection system at different levels of wheel-slip. They conclude that the combination of both probabilistic and deterministic methods lead to more accuracy, particularly if real world factors are becoming significant. Ziparo et al. [48] considered the problem of deploying large robot teams within Urban Search And Rescue (USAR) like environments for victim search. They used RFIDs for coordinating the robots by local search, and extended the approach by a global planner for synchronizing routes in configuration time-space. These approaches are mainly focused on the problem of computing optimal team trajectories in terms of path length and terrain coverage frequency, while there has been only little attention on team robustness. Within real-world scenarios, dynamic changes and system failures are instead crucial factors for any performance metric.

8.3.3 Task Assignment

Cooperation based on Task Assignment has been intensively studied and can be typically considered as a strongly coordinated/distributed approach [19]. In Reactive Task Assignment (e.g., [36]), each member of the team decides whether to engage itself in a task, without re-organizing the other member activities, drastically reducing the requirements on communication but limiting the level of cooperation that they can support. Iterative Task Assignment, such as [27, 46], allocates all tasks present in the system at each time step. In this way, the system can adapt to environmental conditions ensuring a robust allocation, but generally require knowing in advance the tasks that have to be allocated. Sequential Task Assignment methods [23, 49] allocate tasks to robots sequentially as they enter the system, therefore tasks, to be allocated, do not need to be known before the allocation process begins. Such techniques suffer, in general, from a large requirement in terms of bandwidth, due to the large amount of messages exchanged in order to assign tasks. Hybrid solutions which merge characteristics of different types of task allocation have been investigated. For example, in [22] the authors provide an emotion-based solution for multi robot recruitment. Such an approach can be considered intermediate between sequential and reactive task assignment. As previous approaches, this work does not explicitly take into account conflicts due to dynamic task perception. Conflicts arising in Task Assignment are specifically addressed for example by [3, 28]. However, conflicts described in those approaches are only related to the use of shared resources (i.e. space), while other approaches can address a more general class of conflicts, such as the ones that arise when task properties change over time due to dynamic on-line perception [20].

8.3.4 Automatic Video Surveillance

Semi-automated visual surveillance systems deal with the real-time monitoring of persistent and transient objects within a specific environment. The primary aims of these systems are to provide an automatic interpretation of scenes to understand and predict the actions and the interactions of the observed objects based on the information acquired by sensors. As mentioned in [45], the main stages of the pipeline in an automatic visual surveillance system are moving object detection and recognition, tracking and behavioral analysis (or activity recognition). One of the most critical and challenging components of a semi-automated video surveillance system is the low-level detection and tracking phase. Even small detection errors can significantly alter the performance of routines further down the pipeline, and subsequent routines are usually unable to correct errors without using cumbersome, ad-hoc techniques. Adding to this challenge, low-level functions must process huge amounts of data, in real time, over extended periods. This data is frequently corrupted by the camera's sensor (e.g. CCD noise, poor resolution and motion blur), the environment (e.g. illumination irregularities, camera movement, shadows and reflections), and the objects of interest (e.g. transformation, deformation or occlusion). Therefore, to adapt to the challenges of building accurate detection and tracking systems, researchers are usually forced to simplify the problem. It is common to introduce certain assumptions or constraints that may include: fixing the camera [44], constraining the background [43], constraining object movement or applying prior knowledge regarding object-appearance or location [41]. Relaxing any of these constraints often requires the system to be highly application domain oriented. There are two main approaches to object detection: "temporal difference" and "background subtraction". The first approach consists in the subtraction of two consecutive frames followed by thresholding. The second approach is based on the subtraction of a background or reference model and the current image followed by a labeling process. The "temporal difference" has good throughput in dynamic environments as it is very adaptive. However, its performance in extracting all the relevant object pixels is poor. On the other hand, background subtraction approach has a good performance in object extraction. Although, it is sensitive to dynamic changes in the environment; to overcome this issue, adaptive background techniques are applied, which involves creating a background model and continuously upgrading to avoid poor detection in dynamic environments. There are different techniques background modeling, commonly related to the application such as active contours techniques used to track non-rigid objects against homogeneous backgrounds [7], primitive geometric shapes for certain simple rigid objects [16] or articulated shapes for humans in high-resolution images [35]. Background modeling and updating background techniques are based on pixel-based or region-based approaches. In this chapter, an updating technique based on pixel-based background model, Gaussian Mixture Model, GMM [40, 5], for foreground detection in scenario1 is presented. In scenario2, an updating technique based on region-based background model Maximally Stable Extremal Region (MSER) [31] is applied. Moving down in the pipeline of the system after the foreground extraction comes the tracking. Yilmaz et al. [47] reviewed several algorithms, listing the strengths and weaknesses of each of them; emphasizing that each tracking algorithm inevitably fails under certain set of conditions. Therefore, different tracking techniques are used in each scenario as the environment conditions are different in each of them. Therefore, in scenario1 Kalman Filters [26] are implemented and in Scenario 2 a optimized, multi-phased, kd-tree-based [24] tracking algorithm is used. At last, in [10], a survey of activity recognition algorithms is presented where well-known probabilistic approaches, such as Bayesian Networks [11] or Hidden Markov Model [8] are used. In this video surveillance system HMM are used to generalize the object interactions and therefore recognize a predefined activity.

8.4 Representation Formalism

One of our main contribution is the study of a multi-robot patrolling and threat response, with a heterogenous team of agents including both mobile robots and static cameras. The heterogeneity is given not only by the different mobility capabilities of the agents, but also by different sensor abilities. This study is motivated by the fact that integration of many technologies, such as mobile robotics, artificial vision and sensor networks can significantly increase effectiveness of surveillance applications. In such a heterogeneous team, one important issue is to devise a common formalism for representing the knowledge about the environment of the entire system. Our approach to solve the problem of multi-robot monitoring is composed by three components:

1. a map representation of the events occurring in the environment;
2. a generated list of tasks, to handle the events;
3. a coordination protocol, to distribute the tasks among the agents.

The most interesting component is the map representation. Inspired by [29], a Gaussian process models the map to be covered in terms of wide-areas and hot spots. In fact, the map is partitioned in two categories. The objective of the single agent is, then, to cover the assigned areas, prioritizing the hot-spot areas, while keeping the wide-area coverage. In this approach we introduce two novel concepts:

1. we consider different types of events that can occur in the domain at the same time, each one represented with a probabilistic function,
2. we consider decaying of information certainty over time.

Moreover, our system is highly heterogeneous, since it is constituted by both mobile robots carrying different sensors and static cameras. The proposed formalism, thus, allows for a unified representation of heterogeneous multi-source exploration of different types of events or threats.

8.4.1 Problem Formulation

Let $\mathscr{X}$ denote a finite set of locations (or cells) in which the environment is divided. This decomposition depends on the actual environment, robot size, sensor capabilities and event to be detected. For example, in our experimental setup, we monitor an indoor environment looking for events related to people moving around and unattended luggage, and we use a discretization of the ground plane of $20 \times 20cm$. Let $\mathscr{E}$ denote a finite set of events that the system is required to detect and monitor. Let $\mathscr{Z}$ denote a finite set of sensors included in the systems: they can be either fixed or mounted on mobile platforms. For each event $e \in \mathscr{E}$ there is a probability (or belief) that the event is currently occurring at a location $x \in \mathscr{X}$; this probability is denoted by $P_e(x)$. This probability distribution obviously sums to 1 (i.e., $\Sigma_x P_e(x) = 1$). This means that we assume that an event e is occurring (even if this is not the case), and that the team of agents performs a continuous monitoring of the environment. In other words, when a portion of the environment is examined and considered to be clear (i.e., low values of $P_e(x)$), then in another part of the environment that is not examined this probability increases and thus it will become the objective of a next search. It is also to be noted that this representation is adequate when the sensors cover only a part of the environment at any time, as in our setting, while it is not feasible in cases where sensors cover the entire environment.

The computation of this probability distribution is performed by using sensors in $\mathscr{Z}$. Given a sensor $z \in \mathscr{Z}$ and a set of sensor readings $z_{0:t}$ from time 0 to current time t, the probability that event e occurs in location x at time t can be expressed as

$$p_e(x_t|z_{0:t}) = \eta\, p_e(z_t|x_t) \int_{x_{t-1}} p_e(x_t|x_{t-1}) p_e(x_{t-1}|z_{0:t-1}) dx_{t-1} \tag{8.1}$$

Equation 8.1 is derived from Bayes Theorem (see for example [42]). The set of probability distributions $p_e(x_t|z_{0:t})$ for each event e represents a common formalism for the heterogeneous team considered in this work and allows for both driving the patrolling task and evaluating different strategies. This representation has an important feature: it allows for explicitly defining a sensor model for each pair sensor,event. In fact, $p_e(z_t|x_t)$ represents the sensor model of sensor z in detecting event e. In this way, it is possible to accurately model heterogeneous sensors within a coherent formalism. Also the motion model $p(x_t|x_{t-1})$ can be effectively used to model both static objects (e.g., bags) and dynamic objects (e.g., persons). It is important to notice also that the sensor model $p_e(z_t|x_t)$ contributes all the cells x_t that are actually observed by the sensor and to cells that are not within its field of view. In this latter case, the probability of presence of the event is set to the nominal value λ and thus $P_e(x_t)$ tends to λ (i.e., no knowledge) if no sensor is examining cell x_t for some time. This mechanism implements a form of decay of information over time and requires the agents to continuously monitor the environment in order to assess that no threats are present. Usually, the *idleness* [14] is normally used in evaluating multi-agent patrolling approaches. This concept can be extended to our formalization as follows. Given a minimum value of probability γ, that can de defined according to the sensor models for all the events, the *idleness* $I_e(x,t)$ for an event e of a location x at time t is defined as the time elapsed since the location had a low (i.e. $< \gamma$) probability of hosting the event. More formally

$$I_e(x,t) = t - \hat{t} \text{ such that } p_e(x_{\hat{t}}) < \gamma \wedge \forall \tau > \hat{t},\ p_e(x_\tau) \geq \gamma$$

Then the *worst idleness* $WI_e(t)$ for an event e at time t is defined as the biggest value of the idleness for all the locations. Formally

$$WI_e(t) = \max_x I_e(x,t)$$

8.5 Event-Driven Distributed Monitoring

As stated before, we consider two different classes of tasks.

Patrolling tasks define areas of the environment that should be traveled regularly by the agents. The shape of these areas is not constrained to be any specific one. We assume, however, that a decomposition can be performed, to apply standard approaches to coverage problem [15].

Threat response tasks specify restricted portions of the map, where potentially dangerous events are currently occurring. The kind of threat is left unspecified, since it is dependent on the application domain. Examples of considered events are: an intruder detected in a restricted area, a bag or an unknown object left in a clear zone, a non-authorized access to a controlled room. The appropriate response, then, should be specified per application. We assume that the basic response for all these events requires for the agent to reach the location on the map. In this sense, they are the hot-spots specified in 8.4.1.

8.5.1 Layered Map for Event Detection

Figure 8.2 shows a diagram of the proposed solution. Data acquired by the sensors of the system are first locally filtered by each agent and then globally fused in order to determine a global response to perceived events. A finite set of event maps models the event space $\mathscr{E}$. For each sensor in the set $\mathscr{S}$, it is possible to define a sensor model. Each sensor model defines a probability distribution function (pdf) that describes how the sensor perceives an event, its uncertainty and how to update the event map. A sensor can update different event maps, and, hence, it becomes important to define how heterogeneous sensors update the map. A *Layered Map for Event Detection* defines a multi-level Bayesian filter. Each level describes a probability distribution related to a specific sensor: the combination of several levels results in

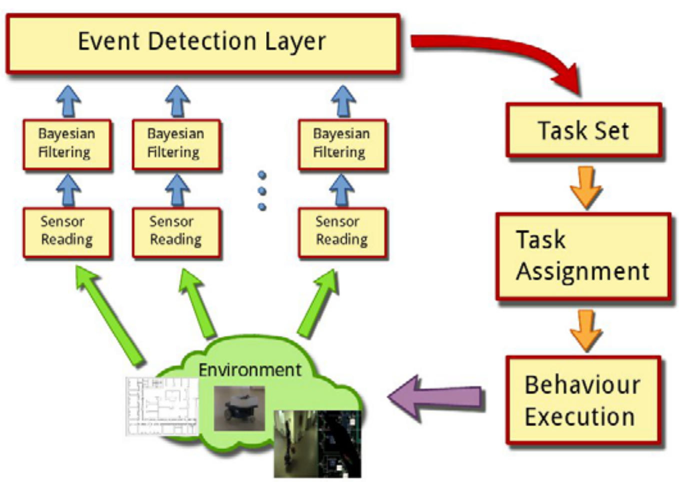

Fig. 8.2 The data flow in the Layered Map for Event Detection approach.

a probability distribution for the event of interest. However, the importance of an event decays when the time goes by: to reflect the temporal constraints in the event handling, we introduced an *aging update step*. This step acts before updating the filter, given the observation from sensors (like in the Predict step of recursive Bayesian filters). The pdf associated to each sensor level has this meaning:

$$p(x) = \begin{cases} 1 & \text{if the sensor filter has converged in a hot-spot} \\ 0 & \text{if there is no relevant information given by the sensor in } x \end{cases}$$

This $p(x) = 0.5$ means that, in that point, the sensor has complete ignorance about the environment it can perceive. Given this assumptions, in every time frame, the pdf of sensor level smooths towards *complete ignorance*:

$$p(x) = \begin{cases} p(x) + \delta_{increase} & \text{if } p(x) < 0.5 \\ p(x) - \delta_{decrease} & \text{if } p(x) > 0.5 \end{cases}$$

The combination of the sensor level is demanded to the Event Detection layer. This layer has a bank of filters, each one delegated to detect a specific event. Each filter uses the belief from a subset of the sensor levels to build a joint belief, representing the pdf of associated event. The characterization of the event depends on the behavior an agent can perform to response of it. In this sense, the event is a *threat* and the response to it depends on the coordination step presented in Section 8.5.2. This process is formalized in Algorithm 8.1.

Algorithm 8.1. MSEventDetect

---

input : u = action performed by the agent
z_s = set of sensor reading from a specific sensor
$\mathbf{BF}$ = set of Bayesian sensor filters
output : $\mathscr{E}$ = set of pdf associated to events of interest

// initialize the event belief Bel
$Bel \leftarrow 0$
foreach bf *in* $\boldsymbol{BF}$ **do**
  // apply aging
  $p^{(bf)}(x|z_s) = \begin{cases} p^{(bf)}(x) + \delta_{increase} & \text{if } p^{(bf)}(x) < 0.5 \\ p^{(bf)}(x) - \delta_{decrease} & \text{if } p^{(bf)}(x) > 0.5 \end{cases}$
  // perform Bayesian filtering
  $\texttt{Predict}_{\texttt{bf}_\texttt{i}}(\texttt{u})$
  $\texttt{Update}_{\texttt{bf}_\texttt{i}}(\texttt{z}_\texttt{s})$
  $Bel \leftarrow Bel \cup p^{(bf)}(x)$
end
// build joint belief in the event detection layer $\boldsymbol{D}$
$\mathscr{E} \leftarrow 0$
foreach d *in* $\boldsymbol{D}$ **do**
  $p^{(d)}(x|Bel) = \prod_i \gamma_i bel_i$
  $\mathscr{E} \leftarrow \mathscr{E} \cup p^{(d)}(x)$
end

8.5.2 From Events to Tasks for Threat Response

The output of the Event Detection is a distribution over the space of the environment, describing the probability of occurring event in specific areas. The team of agents need to translate these information into tasks to perform. First of all, a clustering is done to extract the high probability peaks of the distribution. The clustering uses a grid-based decomposition of the map, to give a coarse approximation of the distribution itself. If the distribution is multi-modal, then, a cluster will be associated to each peak. Each cluster c_e is then defined in terms of center position and occupancy area: these information will be addressed by the task association. After the list of clusters is generated, a corresponding task list is built. In principle, one task is associated to each cluster. Two categories of tasks are, then, considered: *patrolling* and *threat response*. The super class of *Threat-Response* could comprehend different behaviors: explore the given area with a camera, verify the presence of an intruder or an unexpected object and so on. However, the basic behavior associated to this tasks requires for the agent, to reach the location, or its nearby, and take some kind of action. This means that the Threat-Response category defines a whole class of behaviors, distinguished by the last step. Therefore, in our experimental setup, we consider them as simple behaviors to reach the location, avoiding the specification of other specific actions.

Algorithm 8.2. Event2Task

input : $\mathscr{E}$ = set of pdfs associated to events of interest
output : $\mathscr{T}$ = set of tasks to perform

// *clustering of event set*
$\mathscr{C} \leftarrow 0$
foreach e *in* $\mathscr{E}$ **do**
  $c_e = \texttt{Clusterize(e)}$
  $\mathscr{C} \leftarrow \mathscr{C} \cup c_e$
end
// *associate event to task* $\mathscr{T} \leftarrow 0$
foreach c *in* $\mathscr{C}$ **do**
  $\mathscr{T} \leftarrow \mathscr{T} \cup \begin{cases} patrolling & \text{if } c \in \mathscr{E}_p \\ threat & \text{if } c \in \mathscr{E}_t \end{cases}$
end

Algorithm 8.2 illustrates the steps performed to transform the pdf of an event to a task list. Here, $\mathscr{E}_p$ is the class of events that requires a patrolling task, while $\mathscr{E}_t$ is the class of events requiring a threat-response task. Figure 8.3 shows an example where two pdfs (represented as sets of samples) are processed to obtain two tasks.

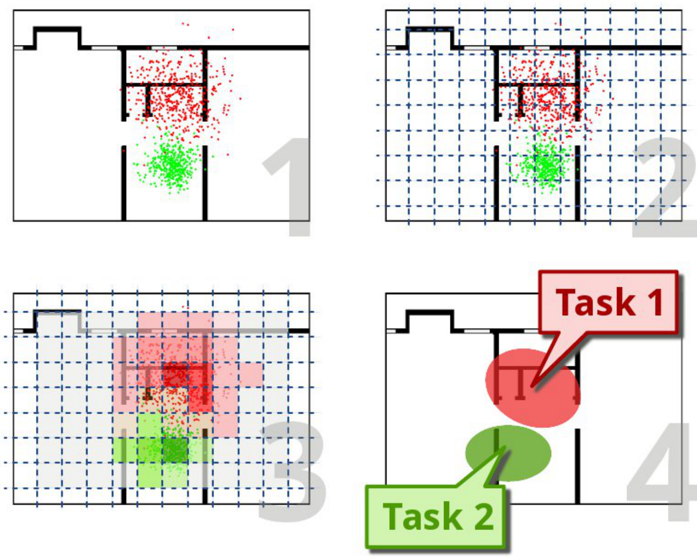

Fig. 8.3 Event to task transformation.

8.5.3 Strategy for Event-Driven Distributed Monitoring

We can now describe the strategy developed for the Event-driven Distributed Monitoring. Algorithm 8.3 incorporates the previously illustrated algorithm for the Event Detection and the event-to-task transformation. The agent starts with a uniform knowledge of the map: no events are detected yet. In normal conditions, the default behavior is to patrol the whole area of the map. At time t, the agent a receives information from the sensors. A sensor can model a real device, as well as a virtual one to describe other types of information (a priori, constraints on the environment and so on).

Algorithm 8.1 is then used to detect a cluster of events on the map. These clusters are then passed to the Algorithm 8.2: a list of tasks is generated. The tasks are spread over the network, to wake up the coordination protocol and the Task Assignment step is performed. Each agent selects the task that is more appropriated to its skills and it signals to other agents its selection. The remaining tasks are relayed to the other agents, that, in the meantime, select the most appropriate task. If the number of tasks is larger than the number of available agents, the non-assigned tasks are put in a queue. When an agent completes its task, this is removed from the pool of tasks for each agent.

Algorithm 8.3. Event Driven Monitoring

input : $\mathbf{BF}$ = set of Bayesian sensor filters
$\mathbf{Z}$ = set of sensor readings from $\mathscr{S}$
$\mathscr{S}$ = set of sensors
M = map of the environment

// initialize the sensor filters
foreach bf *in* $\mathbf{BF}$ **do**
| $p(x)^{(bf)} = \mathscr{U}(M)$
end
// retrieve agent's actions
$u = actions$
// retrieve sensor readings
$\mathbf{Z} \leftarrow 0$
foreach s *in* $\mathscr{S}$ **do**
| $\mathbf{Z} \leftarrow \mathbf{Z} \cup z_s$
end
// detect events
$\mathscr{E}$ = `MSEventDetect`$(u, \mathbf{Z}, \mathbf{BF})$
// generate the task set
$\mathscr{T}$ = `Event2Task`$(\mathscr{E})$
// assign a task to the agent a
$task$ = `TaskAssignment`$(a, \mathscr{T})$

8.6 Implementation and Results

As mentioned in Section 8.2, two scenarios are considered for our system and for each of them different vision algorithms were implemented. In the first scenario, a bag is left unattended and a robot will go and check the suspected area. In the second scenario, the video surveillance system deals with the manipulation of unauthorized objects in a specific positions (laptop in top-left corner in Figure 8.1). The implementation of the computer vision and robotic components to deal with these scenarios and the realization of a full demonstrator to validate the approaches are described in the following.

8.6.1 A Real-Time Multi-tracking Object System for a Stereo Camera - Scenario 1

In this scenario a multi-object tracking algorithm based on a ground plane projection of real-time 3D data coming from a stereo imagery is implemented, giving distinct separation of occluded and closely-interacting objects. Our approach, based on the research activity completed in [25, 26, 33, 34], consists of tracking, using Kalman Filters, fixed templates that are created combining the height and the statistical pixel occupancy of the objects in the scene. These objects are extracted from the background using a Gaussian Mixture Model [40, 5] using four channels: three channels colours (YUV colour space) and a depth channel obtained from the stereo devices [25]. The mixture model is adapted over time and it is used to create a background model that is also upgraded using an adaptive learning rate parameter according to the scene activity level on a per-pixel basis (the value is experimentally obtained). The use of depth information (3D data) can contribute to solve difficult challenges normally faced when detecting and tracking objects such us: improve the foreground segmentation due to its relatively robustness against lighting effects as shadows and also giving the shape feature provides information to discern between people and other foreground objects such as bags. Moreover, the use of a third dimension feature can help to clarify the uncertainty of predictions on the tracking process when an occlusion is produced between a foreground and background object.

The 3D foreground cloud data is then rendered as if it was viewed from an overhead, orthographic camera view (see Figure 8.4); reducing the amount of information and therefore, the computational performance is increased when the tracking is done onto plan-view projection data rather than onto 3D data directly. The projection of the 3D data to a ground plane is chosen due to the assumption that people usually do not overlap in the normal direction to the ground plane. Therefore, this 3D projection allows to separate and to solve some occlusions that more difficult to solve using the original camera-view. The data association implemented in the tracking process is also based on the work presented in [26, 34]. The Gaussian and linear dynamic prediction filters used to track the occupancy and height statistics plan-view maps are the well-known Kalman Filters [9]. Figure 8.5 shows tracking of different types of objects (robot, person and bag), including occlusion. Each plan-view map has been synchronized with its raw frame pair and back projected to the real map of the scene.

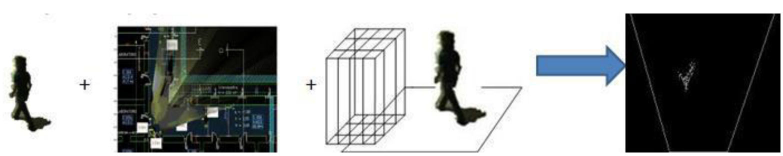

Fig. 8.4 Process for creation of a plan-view

8.6.2 *Maximally Stable Segmentation and Tracking for Real-Time Automated Surveillance - Scenario 2*

In this section we present a novel real-time, color-based, (MSER) detection and tracking algorithm for detecting object manipulation events, based on the work carried out in [21]. Our algorithm synergistically combines MSER-evolution with image-segmentation to produce maximally-stable segmentation. Our MSER algorithm clusters pixels into a hierarchy of detected regions using an efficient line-constrained evolution process. Resulting regions are used to seed a second clustering

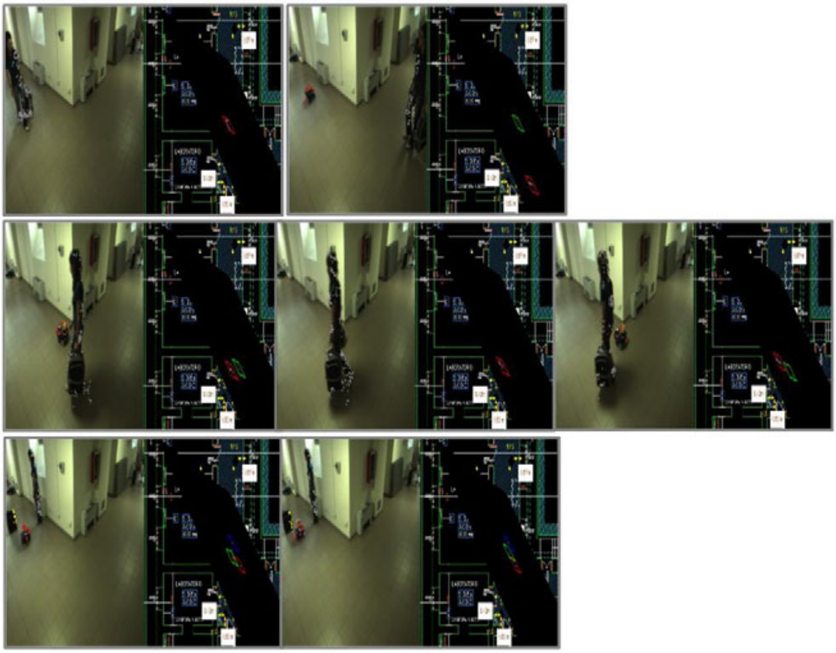

Fig. 8.5 Seven frames of a sequence that shows tracking different types of objects (robot, person and bag), including an occlusion. Each plan-view map has been synchronized with its raw frame pair and back projected to the real plan of the the scene(right side of each image).

process to achieve image-segmentation. The resulting region-set maintains desirable properties from each process and offers several unique advantages including fast operation, dense coverage, descriptive features, temporal stability, and low-level tracking. Regions that are not automatically tracked during segmentation, can be tracked at a higher-level using MSER and line-features. We supplement low-level tracking with an algorithm that matches features using a multi-phased, kd-search algorithm. Regions are modeled and identify using transformation-invariant features that allow identification to be achieved using a constant-time hash-table. To demonstrate the capabilities of our algorithm, we apply it to a variety of real-world activity-recognition scenarios. MSER algorithm is used to reduce unimportant data, following Mikolajczyk [32] final conclusions on comparison of the most promising feature-detection techniques. The MSER algorithm was originally developed by Matas et al. [31] to identify stable areas of light-on-dark, or dark-on-light, in greyscale images. The algorithm is implemented by applying a series of binary thresholds to an image. As the threshold value iterates, areas of connected pixels grow and merge, until every pixel in the image has become a single region. During this process, the regions are monitored, and those that display a relatively stable size through a wide range of thresholds are recorded. This process produces a hierarchical tree of nested MSERs. The tree-root contains the MSER node that comprises every pixel in the image, with incrementally smaller nested sub-regions occurring at every tree-branch. The leaves of the tree contain the first-formed and smallest groups of pixels. Unlike other detection algorithms, the MSER identifies comparatively few regions of interest. However, our algorithm returns either a nested set of regions (traditional MSER-hierarchy formation), or a non-nested, non-overlapping set of regions (typical to image segmentation). Using non-nested regions significantly improves tracking speed and accuracy. To increase the number of detections and improve coverage, Forssen [21] redesigned the algorithm to incorporate color information. Instead of grouping pixels based on a global threshold, Forssen incrementally clustered pixels using the local color gradient (i.e. for every pixel p in the image, the color gradient is measured against adjacent pixels p[+] and p[-]). This process identifies regions of similar-colored pixels that are surrounded by dissimilar pixels. In our approach we take advantage of the increased detection offered by Forssen's color-based approach, although in our approach the region growth is constrained using detected lines; improving segmentation results on objects with high-curvatures gradients. To detect lines, the Canny filter is used rather than MSER as it is more effective at identifying a continuous border between objects since it considers a larger section of the gradient. Therefore, our system processes each frame with the Canny algorithm. Canny edges are converted to line-segments and the pixels corresponding to each line-segment is used to constrain MSER growth. Simply speaking, MSER evolution operates as usual, but is not permitted to cross any Canny lines. An example of detected lines is shown in Figure 8.6(Right). Detected lines are displayed in green.

Fig. 8.6 Left: An example of the feed-forward process. Dark-gray pixels are preserved, Light-gray pixels are re-clustered. Center: MSERs are modeled and displayed using ellipses and average color-values. Right: An example of MSER image segmentation. Regions are filled with their average color, detected lines are shown in green, the path of the tracked hand is represented as a red line.

To improve performance on tracking large, textureless objects that are slow-moving or stationary, we apply a feed-forward algorithm, which is a relative simple addition to our MSER algorithm. After every iteration of MSER generation, we identify pixels in the current frame that are nearly identical (RGB values within 1) to the pixel in the same location of the following frame. If the majority of pixels in any given MSER remain unchanged for the following video image, the matching pixels are pre-grouped into a region for the next iteration. This pixel-cluster is then used to seed growth for the next iteration of MSER evolution. Using our feed-forward approach, any region that cannot continually maintain its boundaries, will be assimilated into similarly-colored adjacent regions. After several iterations of region competition, many unstable regions are eliminated automatically without any additional processing (see also Figure 8.6). Once the regions using MSER features and line-corner features are obtained, the tracking algorithm is implemented to operate upon them. Our tracking algorithm applies four different phases to handle a specific type of tracking problem: “Feed-Forward Tracking ”, “MSER-Tracking”, “Line-Tracking”and “Secondary MSER-Tracking”. However, if an object can be tracked in an early phase, later tracking-phases are not applied to the object. By executing the fastest trackers first, we can further reduce resource requirements. In the “Feed-forward Tracking”phase, using our pixel feed-forward algorithm; tracking becomes a trivial matter of matching the pixel’s donor region with the recipient region. In “MSER-Tracking” phase by as mentioned before, eliminating the problem of nesting by reducing the hierarchy of MSERs to non-hierarchical image segmentation, the representation becomes a one-to-one correspondence and matches are identified using a greedy approach. The purpose of this phase of tracking is to match only those regions that have maintained consistent size and color between successive frames. Each image region is represented by the following features: Centroid (x,y) image coordinates, Height and Width (second-order moment of pixel-positions) and finally color values. Matching is only attempted on regions that remained un-matched after the “Feed-Forward Tracking”phase. Matches are only assigned when regions have similarity measures beyond a predefined similarity threshold. In this tracking phase, line-corners are matched based on their positions, the angles of the associated lines, and the colors of the associated regions. It should be mentioned that, even if a line separates (and is therefore associated with) two regions, that line will have different properties for each region. Specifically, the line angle will be 180 degrees rotated from one region to the other, and the left and right endpoints will be reversed. Each line-end is represented by the following features: Position (x,y) image coordinates, Angle of the corresponding line, RGB color values of the corresponding region and Left / Right handedness of endpoint (Perspective of looking out from the center of the region). Line-corner matching is only attempted on regions that remained un-matched after the “MSER-Tracking”phase. At last, on the “Secondary MSER-Tracking”phase a greedy approach is used to match established regions (regions that were being tracked but were lost) to unassigned regions in more recent frames. Unlike the first three phases, which only consider matches between successive frames, the fourth phase matches regions within an n-frame window. Although there may be several ways to achieve foreground detection using our tracking algorithms, we feel it would be appropriate to simply follow the traditional pipeline. To this affect, the first several frames in a video sequence are committed to building a region-based model of the background. Here, MSERs are identified and tracked until a reasonable estimation of robustness and motion can be obtained. Stable regions are stored to the background model using the same set of features listed in the tracking section. Bear in mind that since background features are continually tracked, the system is equipped to identify unexpected changes to the background. The remainder of the video is considered the operation phase. Here, similarity measurements are made between regions in the background model, and regions found in the current video frame. Regions considered sufficient dissimilar to the background are tracked as foreground regions. Matching regions are tracked as background regions.

Once the foreground is segmented from the background, a color and shape-based model is generated from the set of foreground MSER features. Our technique uses many of the principals presented by Chum and Matas[13], but our feature vectors were selected to provide improved robustness in scenes where deformable or unreliable contours are an issue. Therefore, we propose an algorithm that represents objects using an array of features that can be classified into three-types: MSER-pairs (a 4-dimensional feature-vector), MSER-individuals (a 3-dimensional feature-vector) and finally, Size-position measure (a 2-dimensional feature-vector), feature-set only used for computing vote-tally.

The recognition of activities is done without sophisticated algorithms (a Hidden Markov Model was used to generalize object interactions), so for this surveillance system, we use a simple rule-sets based only on proximity and trajectories. Figure 8.7 shows an example of activity recognition using MSER. Each object is associated by a color bar at the right of the image. The apparent height of the bar corresponds to the computed probability that the person's hand is interacting with that object. In the scenario shown on the left, a person engaged in typical homework-type behaviors including: typing on a laptop; turning pages in a book; moving a mouse; and drinking from a bottle. In the scenario on the right, a person reached into a bag of chips multiple times, and extinguished a trash-fire with a fire extinguisher.

Fig. 8.7 An example of activity recognition using MSER

8.6.3 Multi-robot Environmental Monitoring

The implementation of the Multi-Robot Environmental Monitoring described in Section 8.5 has been implemented on a robotic framework and tested both on 2 Erratic robots[2] and on many simulated robots in the Player/stage environment[3].

---
[2] www.videre.com
[3] playerstage.sourceforge.net

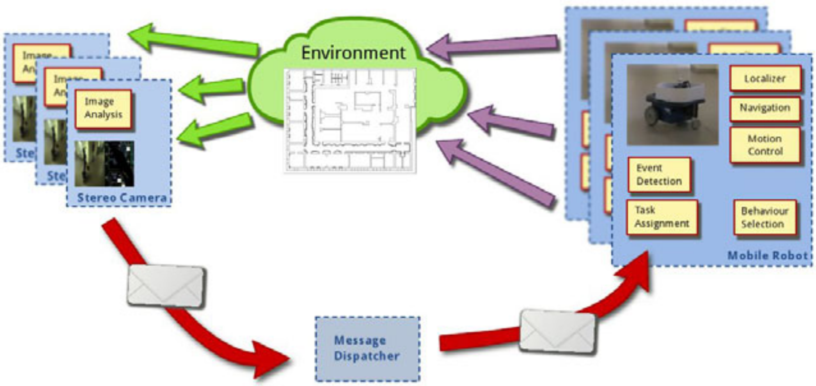

Fig. 8.8 Block diagram of proposed architecture.

Figure 8.8 shows the block diagram of the overall system and the interactions among the developed modules. In particular, the team of robots monitors the environment while waiting for receiving event messages from the vision sub-system.

As previously described, we use a Bayesian Filtering method to achieve the Sensor Data Fusion. In particular, we use a Particle Filter for the sensor filters and event detection layer. In this way, the pdf describing the belief of the system about the events to be detected are described as sets of samples, providing a good compromise between flexibility in the representation and computational effort.

The implementation of the basic robotic functionalities and of the services needed for multi-robot coordination is realized using the OpenRDK toolkit[4] [12]. The mobile robots used in the demonstrator have the following features:

- Navigation and Motion Control based on a two-level approach: a motion using a fine representation of the environment and a topological path-planner that operates on a less detailed map and reduces the search space; probabilistic roadmaps and rapid-exploring random trees are used to implement these two levels [13].
- Localization and Mapping based on a standard particle filter localization method and a well-known implementation GMapping[5] that has been successfully experimented on our robots also in other applications [30].
- Task Assignment based on a distributed coordination paradigm using utility functions [27] already developed and successfully used in other projects.

Moreover, to test the validity of the approach, we replicate the scenarios in the Player/Stage simulator. defining a map of the real environment used for the experiments, and several software agents, with the same characteristics of the real robots. The combination of OpenRDK and Player/Stage is very suitable to develop and

[4] openrdk.sf.net

[5] openslam.org/gmapping.html

experiment multi-robot applications, since they provide a powerful but yet flexible and easy-to-use robot programming environment.

8.6.4 System Execution

In this section we show the behaviors of the overall surveillance system developed in this project. As stated, the recognition of the predefined activities for the scenarios illustrated in this chapter is done without sophisticated algorithms and simple rule-sets based only on proximity and trajectories are applied.

The communication between the video surveillance system and the multi-robot system is done using a TCP client-server communication interface. Each static stereo camera is attached to a PC computer and they communicate between them and the robots via private wireless network. Each static camera and its PC act as a client and one of this PCs also acts as a server. The PC video-server is the only PC that communicates directly with the robots; the PC video-server becomes a client although when it communicates with the multi-robot system, and then, the robots act

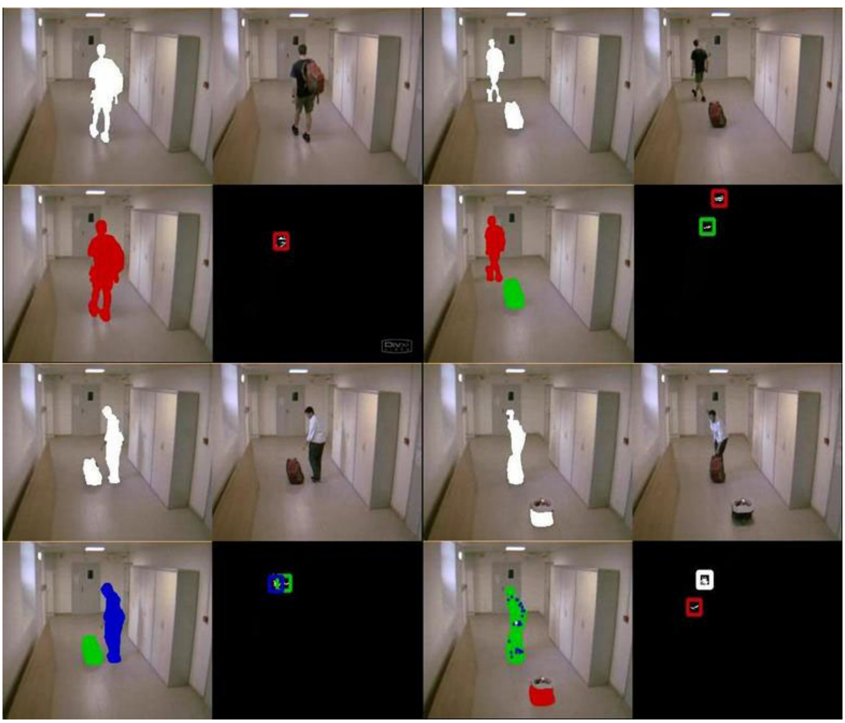

Fig. 8.9 This figure illustrates a sequence of what may happen in Scenario 1. Person A walks through the corridor with the bag and leaves it in the middle of the corridor. Person B approaches the bag and takes it, raising an alarm in the system causing the patrolling robot to go and inspect the area.

Fig. 8.10 This figure illustrates a sequence of what may happen in Scenario 2. Person B places a book(black) and a bottle(green) on the table and manipulates them under the surveillance of the system; until Person B decides to touch an unauthorized object (ie.laptop)(grey) raising an alarm in the system causing the patrolling robot to go and inspect the area.

as servers. Once the video surveillance has recognized an event; e.g. in scenario1, the person associated with the bag abandons the object (see Figure 8.9), the PC client camera sends to the PC video-server the event name and the 3D coordinates. The video-server then constructs a string with this information (first transforms the 3D coordinates to a common coordinate system) and sends the message via wireless to the robots. Then, one of the robots is assigned to go and patrol the area and take a high-resolution picture if the event detected is “bag taken”.

Figures 8.9 and 8.10 show the results in Scenarios 1 and 2 respectively. Figure 8.9 illustrates a sequence of what may happen in Scenario 1. On the top-left image of the figure a person with an object (bag) is walking through the corridor. On the top-right image of the figure, the video system detects that the person left the bag. Therefore a message is sent as “left bag”. On the bottom-left image another person walks very closed to the bag. On the bottom-right image the visual surveillance system detects that a person is taking a bag an a message “bag taken”is sent to the robots and as it can be seen one of the robots is sent to inspect the risen event. Figure8.10 illustrates a sequence of what may happen in Scenario 2. On the top-left, a laptop is placed on the table and one of the robots can be seen patrolling. The top-right and bottom left images of the figure there is a person who is allowed to manipulate different objects. On the bottom-right the person is touching the only object which is not allowed, therefore an alarm "allert"is risen.

8.7 Conclusion

During the recent years, there has been an increased interest in using robotic technologies for security and defence applications, in order to increase their performance and reduce the danger for the people involved. The research proposed in this chapter aims to provide a distributed, multi-robot solution to the problem of environment monitoring, in order to detect or prevent undesired events, such as intrusions, or unattended baggage events or future applications such a fire detection. The problem of detecting and responding to threats through surveillance techniques is particularly well suited to a robotic solution comprising of a team of multiple robots. For large environments, the distributed nature of the multi-robot team provides robustness and increased performance of the surveillance system. In future, the extension of the system by using a group of mobile robots equipped with on-board processing cameras may have several significant advantages over a fixed surveillance camera system. First, the solution could be used in environments that have not been previously engineered with a camera-based monitoring system: the robot team could be deployed quickly to obtain information about a previously unknown environment. Second, the cameras attached to the robots could be moving through the environment in order to best acquire the necessary information, in contrast with a static camera, which can only perform observations from a fixed view point. Third, the robots in the team have the power to collaborate on the monitoring task and are also able to take actions that could pre-empt a potential threat. Fourth, the robots could be equipped with additional specialized sensors, which could be delivered at the appropriate place in the environment to detect the presence of chemical or biological agents. Last, the robot team could communicate with a human operator and receive commands about the goals and potential changes in the mission, allowing for a dynamic, adaptive solution.

Acknowledgements

This publication was developed under Department of Homeland Security (DHS) Science and Technology Assistance Agreement No. 2009-ST-108-000012 awarded by the U.S. Department of Homeland Security. It has not been formally reviewed by DHS. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the U.S. Department of Homeland Security. The Department of Homeland Security does not endorse any products or commercial services mentioned in this publication.

References

[1] Agmon, N.: Multi-robot patrolling and other multi-robot cooperative tasks: An algorithmic approach. Ph.D. thesis, BarIlan University (2009)
[2] Agmon, N., Kraus, S., Kaminka, G.: Multi-robot perimeter patrol in adversarial settings. In: Proc. of IEEE International Conference on Robotics and Automation (ICRA), pp. 2339–2345 (2008)
[3] Alami, R., Fleury, S., Herrb, M., Ingrand, F., Robert, F.: Multi robot cooperation in the martha project. IEEE Robotics and Automation Magazine 5(1), 36–47 (1998)
[4] Almeida, A., Ramalho, G., Santana, H., Tedesco, P.A., Menezes, T., Corruble, V., Chevaleyre, Y.: Recent advances on multi-agent patrolling. In: Bazzan, A.L.C., Labidi, S. (eds.) SBIA 2004. LNCS (LNAI), vol. 3171, pp. 474–483. Springer, Heidelberg (2004)
[5] Bahadori, S., Iocchi, L., Leone, G.R., Nardi, D., Scozzafava, L.: Real-time people localization and tracking through fixed stereo vision. Applied Intelligence 26, 83–97 (2007)
[6] Basilico, N., Gatti, N., Rossi, T., Ceppi, S., Amigoni, F.: Extending algorithms for mobile robot patrolling in the presence of adversaries to more realistic settings. In: WI-IAT 2009: Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology, pp. 557–564. IEEE Computer Society Press, Washington (2009)
[7] Baumberg, A., Hogg, D.C.: Learning deformable models for tracking the human body. In: Shah, M., Jain, R. (eds.) Motion-Based Recognition, pp. 39–60 (1996)
[8] Brand, M., Oliver, N., Pentland, A.: Coupled hidden markov models for complex action recognition. In: CVPR '97: Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR 1997), p. 994. IEEE Computer Society Press, Washington (1997)
[9] Brown, R., Hwang, P.: Introduction to Random Signals and Applied Kalman Filtering. John Wiley & Sons, Chichester (1997)
[10] Buxton, H.: Generative models for learning and understanding dynamic scene activity. In: ECCV Workshop on Generative Model Based Vision, pp. 71–81 (2002)
[11] Buxton, H., Gong, S.: Advanced visual surveillance using bayesian networks. In: International Conference on Computer Vision, pp. 111–123 (1995)
[12] Calisi, D., Censi, A., Iocchi, L., Nardi, D.: OpenRDK: a modular framework for robotic software development. In: Proc. of Int. Conf. on Intelligent Robots and Systems (IROS), pp. 1872–1877 (2008)
[13] Calisi, D., Farinelli, A., Iocchi, L., Nardi, D.: Autonomous navigation and exploration in a rescue environment. In: Proceedings of the 2nd European Conference on Mobile Robotics (ECMR), pp. 110–115 (2005)
[14] Chevaleyre, Y.: Theoretical analysis of the multi-agent patrolling problem. In: IAT 2004: Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology, pp. 302–308. IEEE Computer Society, Washington (2004)
[15] Choset, H.: Coverage for robotics - a survey of recent results. Ann. Math. Artif. Intell. 31(1-4), 113–126 (2001)
[16] Comaniciu, D., Ramesh, V., Meer, P.: Kernel-based object tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence 25(5), 564–575 (2003)
[17] Correll, N., Martinoli, A.: Robust distributed coverage using a swarm of miniature robots. In: Proc. of IEEE International Conference on Robotics and Automation (ICRA), pp. 379–384 (2007)

[18] Elmaliach, Y., Agmon, N., Kaminka, G.A.: Multi-robot area patrol under frequency constraints. In: Proc. of IEEE International Conference on Robotics and Automation (ICRA), pp. 385–390 (2007)
[19] Farinelli, A., Iocchi, L., Nardi, D.: Multi robot systems: A classification focused on coordination. IEEE Transactions on System Man and Cybernetics, part B 34(5), 2015–2028 (2004)
[20] Farinelli, A., Iocchi, L., Nardi, D., Ziparo, V.A.: Assignment of dynamically perceived tasks by token passing in multi-robot systems. Proceedings of the IEEE 94(7), 1271–1288 (2006)
[21] Forssén, P.E.: Maximally stable colour regions for recognition and matching. In: IEEE Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, IEEE, Minneapolis, USA (2007)
[22] Gage, A., Murphy, R.R.: Affective recruitment of distributed heterogeneous agents. In: Proc. of Nineteenth National Conference on Artificial Intelligence, pp. 14–19 (2004)
[23] Gerkey, B., Mataric, M.J.: Principled communication for dynamic multi-robot task allocation. In: Proceedings of the Int. Symposium on Experimental Robotics, pp. 353–362 (2000)
[24] Goodman, J., O'Rourke, J.: Nearest neighbors in high dimensional spaces. In: Piotr Indy, K. (ed.) Handbook of Discrete and Computational Geometry, 2nd edn. IEE Professional Applications of Computing Series, vol. 5, ch. 39 (2004)
[25] Harville, M., Gordon, G., Woodfill, J.: Foreground segmentation using adaptive mixture models in color and depth. In: IEEE Workshop on Detection and Recognition of Events in Video, vol. 0, p. 3 (2001)
[26] Harville, M., Li, D.: Fast, integrated person tracking and activity recognition with plan-view templates from a single stereo camera. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 398–405 (2004)
[27] Iocchi, L., Nardi, D., Piaggio, M., Sgorbissa, A.: Distributed coordination in heterogeneous multi-robot systems. Autonomous Robots 15(2), 155–168 (2003)
[28] Jung, D., Zelinsky, A.: An architecture for distributed cooperative planning in a behaviour-based multi-robot system. Journal of Robotics and Autonomous Systems 26, 149–174 (1999)
[29] Low, K.H., Dolan, J., Khosla, P.: Adaptive multi-robot wide-area exploration and mapping. In: Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008), pp. 23–30 (2008)
[30] Marchetti, L., Grisetti, G., Iocchi, L.: A comparative analysis of particle filter based localization methods. In: Lakemeyer, G., Sklar, E., Sorrenti, D.G., Takahashi, T. (eds.) RoboCup 2006: Robot Soccer World Cup X. LNCS (LNAI), vol. 4434, pp. 442–449. Springer, Heidelberg (2007)
[31] Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust wide baseline stereo from maximally stable extremal regions. In: Proc. of British Machine Vision Conference, vol. 1, pp. 384–393 (2002)
[32] Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Kadir, T., Gool, L.V.: A comparison of affine region detectors. International Journal of Computer Vision 65 (1-2), 43–72 (2005)
[33] Mu noz-Salinas, R., Aguirre, E., García-Silvente, M.: People detection and tracking using stereo vision and color. Image Vision Comput. 25(6), 995–1007 (2007)
[34] Mu noz-Salinas, R., Medina-Carnicer, R., Madrid-Cuevas, F.J., Carmona-Poyato, A.: People detection and tracking with multiple stereo cameras using particle filters. J. Vis. Comun. Image Represent. 20(5), 339–350 (2009)

[35] Ning, H.Z., Wang, L., Hu, W.M., Tan, T.N.: Articulated model based people tracking using motion models. In: Proc. Int. Conf. Multi-Model Interfaces, pp. 115–120 (2002)
[36] Parker, L.E.: ALLIANCE: An architecture for fault tolerant multirobot cooperation. IEEE Transactions on Robotics and Automation 14(2), 220–240 (1998)
[37] Portugal, D., Rocha, R.: Msp algorithm: multi-robot patrolling based on territory allocation using balanced graph partitioning. In: SAC 2010: Proceedings of the 2010 ACM Symposium on Applied Computing, pp. 1271–1276. ACM, New York (2010)
[38] Qu, Y.G.Z.: Coverage control for a mobile robot patrolling a dynamic and uncertain environment. In: WCICA 2004: Proceedings of 5th World Congress on Intelligent Control and Automation, pp. 4899–4903 (2004)
[39] Sak, T., Wainer, J., Goldenstein, S.K.: Probabilistic multiagent patrolling. In: Zaverucha, G., da Costa, A.L. (eds.) SBIA 2008. LNCS (LNAI), vol. 5249, pp. 124–133. Springer, Heidelberg (2008)
[40] Stauffer, C., Grimson, W.E.L.: Adaptive background mixture models for real-time tracking. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 246–252 (1999)
[41] Tan, T.N., Sullivan, G.D., Baker, K.D.: Model-based localization and recognition of road vehicles. International Journal Computer Vision 29(1), 22–25 (1998)
[42] Thrun, S., Burgard, W., Fox, D.: Probabilistic Robotics. The MIT Press, Cambridge (2005)
[43] Tian, T., Tomasi, C.: Comparison of approaches to egomotion computation. In: Computer Vision and Pattern Recognition, pp. 315–320 (1996)
[44] Toyama, K., Krumm, J., Brumitt, B., Meyers, B.: Wallflower: Principles and practice of background maintenance. In: IEEE International Conference on Computer Vision, vol. 1, p. 255 (1999)
[45] Valera, M., Velastin, S.A.: chap. 1. A Review of the State-of-the-Art in Distributed Surveillance Systems. In: Velastin, S.A., Remagnino, P. (eds.) Intelligent Distributed Video Surveillance Systems. IEE Professional Applications of Computing Series, vol. 5, pp. 1–25 (2006)
[46] Werger, B.B., Mataric, M.J.: Broadcast of local eligibility for multi-target observation. In: DARS 2000, pp. 347–356 (2000)
[47] Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. ACM Computer Survey 38, 13 (2006)
[48] Ziparo, V., Kleiner, A., Nebel, B., Nardi, D.: Rfid-based exploration for large robot teams. In: Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA), Rome, Italy (2007)
[49] Zlot, R., Stenz, A., Dias, M.B., Thayer, S.: Multi robot exploration controlled by a market economy. In: Proc. of IEEE International Conference on Robotics and Automation (ICRA), pp. 3016–3023 (2002)