# Multi-Agent Surveillance and Threat Evaluation for Indoor Environment

**Ali Nasir, Adeel Arif**
Electrical Engineering Department University of Central Punjab Lahore, Pakistan

***Abstract***

This paper discusses a hierarchical multi-agent architecture for implementing semi-automated surveillance in the indoor environment. Specifically, a scientific lab environment is considered for demonstration purposes. A three-layered multiagent architecture of the surveillance system is presented with local intelligence, global intelligence, and human supervision. The internal architectures, models, and algorithms for local and global intelligence agents are presented and discussed in the context of surveillance of a scientific laboratory. Possible issues in practical implementation, scaling, and advancement of the proposed approach are also discussed in the paper.

***Keywords***

*Multiagent systems; hierarchical decision making; video surveillance*

## I. Introduction

The trend of video surveillance has grown rapidly due to the increasing threats to public safety. In this regard, millions of security cameras have been installed in different countries. Pakistan government is also realizing the importance of video surveillance and in this regard, more than 1800 surveillance cameras have been installed recently in Islamabad under the “safe city” project. Thousands of cameras are to be installed in other major cities of Pakistan as well. We are past that point where we discuss whether we should have surveillance cameras or not. The major question to ask right now is how to make the surveillance system efficient and effective.

The traditional way of surveillance where guards watch live footage on screens in a control room is no longer feasible. For example 1800 cameras in Islamabad require 450 guards to monitor (assuming 1:4 ratio for cameras versus guards). Ideally, 1800 screens are also required but if the view is split screened, maybe 450 screens could be used (4 views per screen). Even then, each guard has to take a small five to ten minute break in each hour for maintenance of health. Timely response and identification of threat is difficult in such circumstances. Therefore aid of technology (especially artificial intelligence) is required to assist traditional surveillance system.

Fortunately there has been plenty of work on video surveillance in different areas e.g. object detection, object tracking, behavioral analysis, and event detection [5][6]. Surveillance in particular has acquired the attention of researchers for a long time now. Most of the literature on surveillance is on surveillance of public places and outdoor environment [1][2]. Specifically, surveillance problem using drones (quadrotors or fixed wing unmanned aerial vehicles) has been addressed by [1][2][4] and many others. Multiagent surveillance with distributed intelligence has been proposed by [7] where one major point to ponder is whether the information from various cameras shall be merged by each of the “agents” or only some agents. Furthermore, which agent should use which information (or whether all agents should use all information) is an important issue to address especially when the available memory and computational power is limited. This issue is somewhat addressed in [7] by classifying information to be communicated as “relevant” and “irrelevant”. But there is another systematic way of addressing the issue of information handling and that is to have a supervisory agent with superior computational and data storage capability.

In this paper, we propose a novel idea and algorithms for indoor video multiagent surveillance. The novelty in the approach is in the use of three-layer architecture resulting in hierarchical multiagent surveillance system. Furthermore, the approach is tailored for the requirements and constraints of indoor environment as opposed to much generic approach found in existing literature. Major benefit of focusing on indoor problems is that the space of possibilities is much more limited as compared to an open or public area. Due to this, we can afford to use tools in artificial intelligence that are computationally expensive and hard to implement on problems with large state space.

## II. The Proposed Architecture

### *A. System Level Architecture*

The problem of surveillance in general is too broad to be addressed in one paper. There are many aspects of surveillance that form proper research areas such as activity analysis, object (or face) detection and identification, object (or human) tracking, camera calibration and multi-camera topology definition [6]. Most of the algorithms involved in such research involve image processing and artificial intelligence. This paper is concerned with the artificial intelligence part of the problem where inference about the situation (or scene) is drawn based on the processed image data. Specifically, we propose a multiagent approach for autonomous surveillance. A multiagent approach has been discussed in the past [3].

Approach in [3] presents a more generic perspective and the use of multiagent is mainly to distribute the intelligence and hence the computational burden. The approach we propose however is intended for more than just distribution of intelligence, we also divide the intelligence into two hierarchical levels i.e. local intelligence and global intelligence. Fig. 1 shows the basic architecture of the surveillance system that represents the proposed hierarchical multiagent intelligence. Fundamentally, the decision making and intelligence in the proposed system is at three levels where two levels are autonomous and the highest level is manual involving human security officials. This is one other unique feature of our proposed approach i.e. the intelligence involved is not entirely that of computerized software. Having security officials can save complex decisions and remove uncertainties which otherwise would be hard to deal with. The proposed surveillance system works in the following manner. Image or raw data is gathered by the cameras. The data is converted into digital form and forwarded to the local intelligence algorithms for further processing. The local intelligence agents inspect the data for presence of any of the predefined features of interest. All information obtained at local agents is forwarded to the global agent where rigorous analysis is performed to correlate information and determine the overall situation. After processing the local information, the global agent generates alarms or warnings for the security officials and low level processing commands for the local agents. Low level processing commands are intended for further elaboration on a specific feature of interest. The security officials act according to the alarms and warnings and update the global agent about the situation e.g. threat taken care of or threat found to be nonexistent. Global agent can also learn based on the feedback from the security officials in response to specific alarms and warnings.

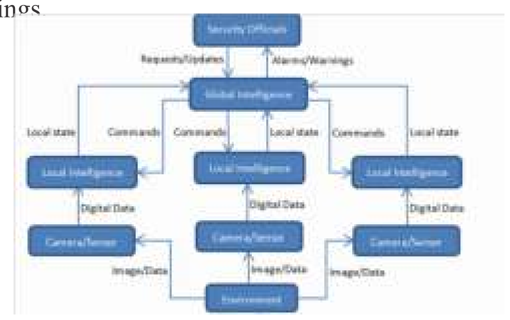

Fig. 1: Surveillance System Overview

The concept of hierarchical multiagent surveillance described above has certain advantages over the conventional surveillance and the multiagent system proposed in [3]. For example due to the hierarchy, the computation at the local agents is simplified because each agent has to process information obtained by its own designated sensors only. This is useful advantage since each local agent may not have infinite backup power and it is usually not feasible to install big computing devices at each local agent location. On the other hand, global agent could be a state of the art computer placed in the control room with ample backup power to ensure uninterrupted operation of the system. Furthermore, since all agents communicate with one global agent, the communication equipment required to implement the system is also simplified. Finally the involvement of human intelligence layer saves the global agent from complex detailed analysis that is difficult to do using the limited information obtained from the local agents. It is relatively easier for a computerized agent to detect an ambiguity whereas determination of the exact nature of ambiguity may not be easy to find in all situations. In such cases, the global agent can generate a warning and the human security officials can confirm or reject the presence of a threat. Based on human feedback, the global agent can learn about various situations thus improving the efficiency of the system.

### *B. Agent Level Architecture*

At the agent level, we propose the use of three-tier computational intelligence architecture [8] with some modifications to suit the surveillance application. The original three-tier is intended for spacecraft operations with three layers called the controller, the sequencer, and the deliberator. In surveillance, we need different architectures for local and global agents. For the local agents, the computational layers are the object detection, object identification, and the activity analysis layer. Fig. 2 shows the three-tier architecture for local agent. Note that each agent may have different routines specific to the detection of various objects. Similarly, object identification and activity analysis may be done using more than one scheme. Purpose of having multiple methods is that not all objects and activities can be identified using a single algorithm.

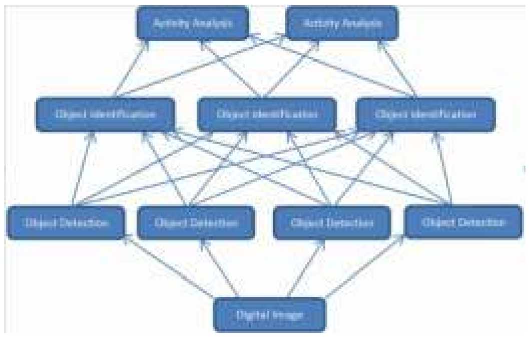

Fig. 2: Local Agent Architecture

Having the three tier architecture proposed above fits smoothly with the surveillance system presented in Fig. 1. Specifically, the commands to the local intelligence shall determine which object detection, identification, and activity analysis algorithms shall be processed. For the global agent, the three-tier architecture is shown in Fig. 3. Note that the three layers here are different i.e. correlation analysis, threat evaluation, and calculation of alarms and commands. Correlation analysis in the global agent is performed to determine the consistency between the states of the cameras with overlapping or adjacent fields of view. Once the correlations analysis is performed, the evaluation of threat is performed. Different threats are evaluated using different algorithms (hence the multiple blocks of threat evaluation).

Finally, the alarms are generated for the threats that are fairly certain (and warnings could be generated for ambiguities). Also generated at the top layer in the global agent are the commands for the local agent to perform a targeted analysis of certain objects or activities.

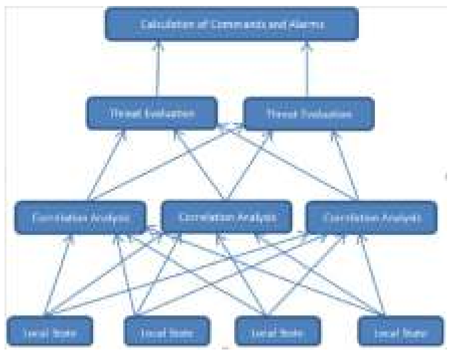

Fig. 3: Global Agent Architecture

## III. Modeling and Algorithms

Before developing algorithms to carry out the tasks involved in different layers presented in the previous section, it is required to develop a mathematical model of the environment. There can be infinitely many possible ways to model an environment. Furthermore a model can be as complex as one desire. But here we stick to a simplest possible model that suffices two objectives. First is the detection of an intruder, and the second is the evaluation of a threat. We assume that the surveillance is to be done inside a research lab. We further assume that all officials working in the lab are known and the timings of the lab are also known. Based on the above assumptions, we consider the following variables in the model for a local agent

$$
\begin{aligned}
&S = \{S_1, S_2, \ldots, S_q\}\\
&S_j = \{O_j, P_j, i_j, w_j, b_j\}\\
&O = \{o_1, o_2, \ldots, o_n\}\\
&o_k \in \{0,1,2\},\ k \in \{1,2,\ldots,n\}\\
&P = \{p_1, p_2, \ldots p_m\}\\
&p_k \in \{0,1\} k \in \{1,2\ldots,m\}\\
&i \in \{0,1,2,3\}\\
&w \in \{0,1,2,3\}\\
&b \in \{0,1,2,3\}
\end{aligned} \tag{1}
$$

In above equation, $S$ represents the set of states of the environment. Each element of the set is a unique combination of the values of all the variables in the model. In our model for a local agent, there are five variables. Variable $O$ is a vector of objects (equipment, furniture etc.) in the lab. Each object $o_j$ can either be missing ($o_j = 0$) or found at its designated place ($o_j = 1$) or found misplaced in the lab ($o_j = 2$). Next variable $P$ is also a vector that represents staff and researchers working in the lab. Each staff member can either be present in the lab ($p_j = 1$) or not present in the lab ($p_j = 0$). Next variable $i$ represents how many (if any at all) intruders are detected in the lab. Note that we have assigned four possible values to the variable $i$. The intention is to differentiate between no intruder ($i = 0$), one or two intruders ($i = 1$), a few intruders ($i$=2), and many intruders ($i$=3). The exact value (or range) of the number of intruders corresponding to "a few" and "many" can be assigned by the user. Next variable in the model is $w$ that represents the presence of weapons. The values assigned to $w$ are to signify type of weapon e.g. small, large, firearm, steel knife etc. In this paper, w = 0 means there are no weapons detected in the lab. Other values of $w$ represent whether the weapon detected in a knife or pointy metal ($w = 1$), a pistol or small size firearm ($w = 2$) or any large size firearm ($w = 3$). Final variable in the model is $b$ representing the kind of activity that might be taking place in the lab. As usual $b = 0$ represents no activity is taking place. Other values of $b$ represent whether normal lab research is taking place ($b = 1$), or there is a hostage situation ($b = 2$), or physical beating ($b = 3$) is going on (or just occurred).

A combination of the variables described above forms local state or state of the environment as far as the local agent is concerned. The state for the global agent has to include information from all local agents and the requests or updates from the security officials. Hence the state can be formulated as

$$
\begin{aligned}
&S = \{s_1, s_2, \ldots, s_r\}\\
&s_j = \{s_{1j}, s_{2j}, \ldots, s_{kj}, G_j\}\\
&s_{vj} = \{O_{vj}, P_{vj}, i_{vj}, w_{vj}, b_{vj}\}\\
&v \in \{1,2,\ldots,k\}, j \in \{1,2,\ldots,r\\
&G = \{g_1, g_2, \ldots, g_f\}\\
&g_l \in \{0,1,2\}, l \in \{1,2,\ldots,f\}
\end{aligned} \tag{2}
$$

Most of the variables in the state of a global agent are the same as those in the local agent states. Vector $G$ is set of variables used to identify requests and information from the security officials. Vector $G$ can expand or shrink as the agent learns more situations or is to forget about some situation. Each variable in vector $g$ can have three possible values corresponding to no information about an event ($g = 0$), a certain event been in hostile situation ($g = 1$) or a certain event been in a nonthreatening situation ($g = 2$). We assume $f$ built-in (or anticipated) events for example: presence of a potential terrorist in the lab, occurrence of a fight in the lab, harmful spill in the lab, presence of weapons in the lab and so on. Note that $g = 0$ is an indication of request for more information on a certain event. Note that the set of alarms that a global agent can trigger is not in the state space. One could have those in the set of action space.

Now we present general algorithms for local and global agents that use the information from the model and perform actions that are required to update the information of the variables involved in the model. Fig. 4 shows algorithm for the local agent. Inputs to the algorithm are the digital image data acquired by the associated camera and the command from the global agent. Local agent is assumed to have a built-in library of images including the images of the lab equipment, lab staff, the placement of the equipment in the lab, possible weapons that could be brought into the lab, and other images that might help in detecting features of interest from the current image acquired. First of all, the local agent addresses any command that is issued by the global agent. This command could be to identify or confirm an intruder or a weapon or it could be to confirm a missing or misplaced equipment etc. The purpose of the command is to perform rigorous image processing on the past and current images in order to identify some feature that is hard to identify or could have been missed by routine algorithms of image processing. This is realistic because routine image processing needs to be fast and to complete a task faster usually requires reduced computations which may lead to some errors in judgement. Since these errors are not frequent, it is worthwhile to have a separate deep search algorithm which executes on demand only.

If there is no specific command from the global agent, the local agent identifies (or updates) the features of the local state using routine algorithms each dedicated for a specific task. For example the compare_image function is for comparing the current image of the lab (or a specific part of the lab) with relevant library images to determine if all objects are present or any (or some) of the objects are missing (or misplaced). If an object of the lab is not supposed to be in the field of view of a certain camera, then it is by-default assumed to be in-place by the local intelligence associated with that camera. To elaborate, let us suppose that a lab has three shelves and three cameras (one for each shelf). If camera 1 does not have any item of shelf 2 and 3 on its image, then local agent 1 will assume that those items are placed on their corresponding shelves whereas local agents 2 and 3 may or may not agree based on what is on their corresponding images. Function for staff detection simply would tell the global agent about which member is in which area of the lab. For instance if a member is not visible to one camera, he/she is absent from lab according to the local intelligence but some camera (assuming that the lab does not have a camera-blind spot) ought to detect that staff member and hence the global intelligence would know about the presence of the same

```
function Local Intelligence (digital_image, Command) returns local
state
inputs: digital_image, Command
static: liberary_images, image_history

if Command not null then do
s ← deep_search (Command, digital_image, liberary_images,
image_history)

else if digital_image not null then do
O ← compare_image (liberary_images, digital_image)
P ← detect_staff (digital_image, liberary_images)
i ← detect_intruder (digital_image, liberary_images)
w ← detect_weapon (digital_image, liberary_images)
b ← detect_activity (digital_image, image_history)
s ← (O, P, i, w, b)
return s
```

Fig. 4: Local Agent Algorithm

The remaining functions can be codded to work in a similar manner. For instance, the intruder detect function can mark the presence of a non-member of the lab as intruder. When this information reaches the global agent, the threat associated with the intruder can be evaluated using predefined algorithms at the global level and by requesting feedback from the security officials (which of course do not have to do any image processing and hence can tell rather quickly just by looking at the live feed whether a person detected as an intruder is worth worrying about or not). Similar intelligence can be used for weapon detection. Note that both intruder detection and weapon detection functions use the image library that includes template images of possible weapons and maybe intruder outfits (or outfits that rule out a person being intruder e.g. a guest badge or sweeper uniform etc.). Finally, the activity detection function serves two purposes. First is to detect activities of any intruders (if any). Second is to detect the activities of the staff and non-intruder personnel. This function is important for a staff member can pose a threat (or someone who may be disguised as one of the staff members or sweeper).

Algorithm for the global agent is shown in Fig. 5. There are three built-in functions in the global agent. Before we discuss the functions, it is important to understand the information contained in the variable (vectors) *camera_topology* and *threat_vector*. The topology vector includes information about the field of views of the local agents or overlap, if any, in the information obtained. This information helps in increasing confidence in the information (assuming that the information is matching rather than conflicting). Topology information also helps avoiding double count of any intruder or weapon. The threat vector is for keeping track of the kinds of threat e.g. unusual activity, presence of weapons or intruders etc. The reason why this vector is a static variable is that a weapon or an intruder cannot just disappear in a flash. So if the local information shows unexpected change, this change can be considered as a threat itself.

```
function Global Intelligence (local_states, security_data) returns
alarms and commands
inputs: local_states, security_data
static: camera_topology, threat_vector

threat_vector ← update_threat_vector (threat_vector, local_states,
camera_topology)
alarms ← generate_alarms (security_data, threat_vector)
for j = 1 to k do
commands(j) ← generate_local_commands (alarms, security_data,
camera_topology)

return alarms, commands
```

Fig. 5: Global Agent Algorithm

The function to update the threat vector is used to determine the changes in the security situation of lab as per new information obtained from all the local agents. This updated threat vector is then used in the alarm generation function that incorporates the security related information obtained from the human officials. The reason to use security data in the alarm generation is to incorporate any inaccuracies in the automated threat evaluation due to image processing errors or other anomalies or special circumstances such as some rehearsal or renovation going on in the lab or presence of armed guards in case of a visit by a high profile person e.g. a minister or a high rank army officer. Finally, the command generation function alarms, security data, and camera topology to determine tasks to be performed by the local agents in case there are any discrepancies, conflicts, or doubts in the information obtained.

## IV. Practical Issues and Advancements in the Proposed Approach

In this section we discuss important issues that can offer nontrivial challenges while implementing our proposed surveillance approach. We also discuss how to advance the architecture in terms of functionality and better adaptability to the ever changing scenarios.

### *A. Computational Complexity*

First we discuss the computational complexity of the problem. As far as the local agents are concerned, most of computations involve image processing but is it really so? The algorithm in Fig. 4, requires the local agent to assign values to each of the state variables. In order to assign one of the possible values, in worst case, the associated function of the algorithm may have to check for possibility of each of the value being true. This means that the larger the domain space of each variable, the larger is the number of computations required to assign its value. Fortunately, since each variable is assigned its value independent of the other variables, the complexity is additive in terms of all variables i.e. the complexity of the local agent is given by

$$C_L = 3^n + 2^m + 4 \times 3 \qquad (3)$$

Here $n$ is the number of objects in the lab and $m$ is the number of staff members. Take an example of the lab with 10 staff members and 50 equipment items. This leads the factor $3^{50}$ as the decisive value of complexity this value is definitely above the capability on the other hand $3^{10}$ or $3^{12}$ is marginally tractable. This means that one camera should not be held responsible for more than 10 to 12 items. Similarly, lab staff cannot exceed a similar limit. This means that in order to keep the complexity of the local agent below the threshold of the technological capability, more agents are to be installed.

Now for global agent, the problem is contradicting because as we increase the number of local agents, the complexity of the global agent increases since the complexity of the global agent is given by

$$C_G = k \times C_L + 3^f \qquad (4)$$

Above equation assumes no correlation between the agents and in such case, the number of agents contribute linearly to the complexity. But if the agents are correlated, then the complexity becomes exponential. For example, assume that out of k local agents, k1 are correlated, then the complexity becomes

$$C_G = (C_L)^{k1} + (k - k1) \times C_L + 3^f \qquad (5)$$

Let us assume we have five local agents (five cameras) and there are 10 events about which the security officials may provide feedback/request to the global agent. First we assume all five local agents to be independent, then the complexity contribution from the local agent and global agent is only different if $3^f$ factor exceeds $3^n$ in the $C_L$. In this situation, as long as the number of events to be handled is small enough, there are no computational problems expected. On the other hand, if three of the agents are correlated, then the complexity due to local agents becomes $3^{3n}$. This indeed can create significant problems and computational delays in the surveillance loop.

Therefore, in order to keep the complexity in the global agent tractable, one need to confine the number of events within a certain limit and also correlation between the agents has to be minimized. Notice here that, the correlation between local agents is useful in terms of robustness in the system and having to avoid correlation to reduce complexity is a tradeoff.

One possible solution to the complexity problem is to have multiple global agents (or to have pseudo-global agents). The idea is to have each pseudo-global agent be assigned a subset of local agents only (or a subset of security-related events only). We leave this idea to be explored in future research on this topic.

### *B. Scalability*

Scalability is important for the approach to be practically usable. Since the paper discusses the approach only for one lab, whereas in real world, there may be tens of labs in a facility. Also there are corridors and lobbies and meeting rooms. Theoretically, if each room or each lobby is secured individually, then the whole building can be considered as secured. But practically, there are certain issues such as exchange of critical information (between global agent of a room and that of a lobby) and the information about who is allowed to access which places.

In principal, the extension of the proposed approach does require nontrivial working in terms of modeling, algorithms, and architecture. On the other hand, the proposed hierarchical approach, the model, and the algorithms serve as a basis for such extensions. The additional features would be required on top of the proposed architecture to facilitate the security of a multi-lab building. For instance, the consideration of which person is among the employees in the building and which member is not. Furthermore, which member belongs in which lab and what equipment belong in which lab. Also, there could be information about restricted versus public areas within the building. The concept of larger scale architecture is shown in Fig. **6**. Here bidirectional arrows indicate two way information exchange and each block labeled as surveillance system comprises of the local and global intelligence agents as in Fig. 1.

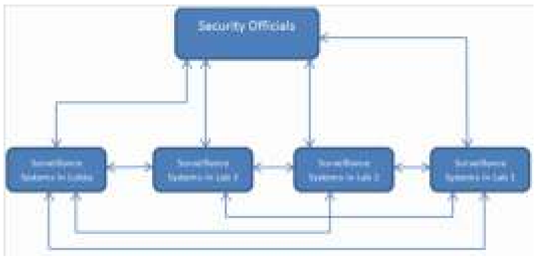
Security Officials
Surveillance Systems in Lobby
Surveillance Systems in Lab 1
Surveillance Systems in Lab 2
Surveillance Systems in Lab 3

Fig. 6: Expansion over the Architecture in Fig. 1

### C. *Online Learning and Updates*

Online learning and updates in the image liberary and threat vector are key ingredients for a long term surveillance solution.

It is not possible to know *all* possible threat types and the images of *all* possible weapons or intruder outfits etc. Many of these things are learnt over time with "experience". Therefore, a natural extension of the proposed architecture is to add learning algorithms in the local and global intelligence agents. Most common form of learning in multiagent systems is reinforcement learning [9]. For the proposed surveillance system, learning can be incorporated at local and global intelligence levels. The local agent can learn to identify the state with better accuracy by using the feedback from human security officials and adding more images to the library of images (or maybe replacing images in some cases). The global agent can learn new threat types and can also learn to add or remove from the set of alarms. Furthermore, the accuracy of alarm generation can be improved through learning that may use feedback from the security officials.

## V. Conclusions

Implementation of intelligent surveillance in general is a problem that cannot be addressed in a single research paper. An attempt has been made to discuss a specific part of the problem i.e. indoor surveillance. In this context, a three-layered multiagent architecture is presented that involves artificial intelligence as well as human supervision. Through the example of a scientific laboratory, it has been demonstrated how the proposed system can be designed. The implementation of the proposed system however is not straight forward for large or medium sized buildings. Most important issue is that of computational complexity. Some ways have been proposed to avoid complexity problems using the insights of the proposed approach. Still a lot of work is to be done in order to mature the idea for practical use. Some directions in this regard have been discussed in the paper.

## References

[1] Kingston, Derek, Randal W. Beard, and Ryan S. Holt. "Decentralized perimeter surveillance using a team of UAVs." *IEEE Transactions on Robotics* 24.6 (2008): 1394-1404.

[2] Varga, Maja, et al. "Evaluation of control strategies for fixed-wing drones following slow-moving ground agents." *Robotics and Autonomous Systems* 72 (2015): 285-294.

[3] Kariotoglou, Nikolaos, et al. "Multi-agent autonomous surveillance: a framework based on stochastic reachability and hierarchical task allocation."*Journal of dynamic systems, measurement, and control* 137.3 (2015): 031008.

[4] Leahy, Kevin, et al. "Persistent surveillance for unmanned aerial vehicles subject to charging and temporal logic constraints." *Autonomous Robots*(2015): 1-16.

[5] Dee, Hannah M., and Sergio A. Velastin. "How close are we to solving the problem of automated visual surveillance?." *Machine Vision and Applications*19.5-6 (2008): 329-343.

[6] Wang, Xiaogang. "Intelligent multi-camera video surveillance: A review."*Pattern recognition letters* 34.1 (2013): 3-19.

[7] Remagnino, Paolo, A. I. Shihab, and Graeme A. Jones. "Distributed intelligence for multi-camera visual surveillance." *Pattern recognition* 37.4 (2004): 675-689.

[8] Gat, Erann. "On three-layer architectures." *Artificial intelligence and mobile robots* 195 (1998): 210.

[9] Wahab, Matthew. "Reinforcement learning in multiagent systems." *McGill university School of computer Science* (2003).