공학석사학위논문

시간간격 구속조건을 갖는 정찰 임무 수행을 위한 혼합 정수계획법 기반의 다중 로봇 협업 기법

Multi-agent Persistent Surveillance with Time-interval Constraints Using Mixed Integer Linear Programming

2014 년 2 월

서울대학교 대학원

기계항공공학부

최 지 은

시간간격 구속조건을 갖는 정찰 임무 수행을 위한 혼합 정수계획법 기반의 다중 로봇 협업 기법

Multi-agent Persistent Surveillance with Time-interval Constraints Using Mixed Integer Linear Programming

지도교수 김 현 진

이 논문을 공학석사 학위논문으로 제출함

2013 년 12 월

서울대학교 대학원

기계항공공학부

최 지 은

최지은의 공학석사 학위논문을 인준함

2013 년 12 월

위 원 장 김 유 단

부위원장 김 현 진

위 원 박 찬 국

to my

MOTHER, FATHER, SISTERS and BROTHER

with love

Multi-agent Persistent Surveillance with Time-interval Constraints using Mixed-integer Linear Programming

by

JIEUN CHOI

Presented to the Faculty of the Graduate School of
Seoul National University
in Partial Fulfillment
of the Requirements
for the Degree of

MASTER OF SCIENCE

School of Mechanical & Aerospace Engineering

Seoul National University

February 2014

Abstract

In the past years, cooperative operation of multiple robots has attracted considerable research interest for coverage and search of broad and complex areas. Unlike the complete coverage problem, in the basic persistent surveillance problem each location in the environment must be visited repeatedly while minimizing the time-interval between any two visits to the same location. In this paper, we propose a cooperative path planning algorithm for an efficient persistent surveillance operation of multiple heterogeneous agents using mixed integer linear programming. Since we specially consider a grid environment with different priories (time interval constraints), agents must visit the region at least once within the specific time interval constraint. The cost function is the maximum risk minimization. Also, we consider a protocol for cooperative movement of heterogeneous agents. The objectives of the proposed algorithm are: persistent surveillance operation with time interval constraints, obstacle avoidance, and collision avoidance among multiple agents. Simulation results confirm that a cell is visited at least once within its time interval constraints.

Table of Contents

List of Tables

List of Figures

1 Introduction

1.1 Previous Works

The persistent surveillance and patrolling problem by multiple agents has received considerable attention in robotics and AI literature recently. Chevaleyre et al.[1] use a multiple Traveling Salesman Problem (TSP) approach to solve the surveillance problem. The authors of [2] introduced an approximate policy for the persistent surveillance problem using a parallel, distributed implementation of approximate dynamic programming. In their problem, an onboard health monitoring system, communication constraints, and stochastic sensor failure are considered. Arvelo et al. [3] set as their objectives finding the minimum number of robots and time-invariant memoryless control policy that guarantees that the largest number of states. In [4], the authors introduced a reactive policy for persistent surveillance with equal visiting frequency using multiple UAVs. In [5], they considered the communication limitations and applied a leader follower interaction protocol(LFIP) based exploration framework. Pasqualetti et al.[6] applied graph partition approaches to the patrolling problem. Market-based auction algorithm can be applied to persistent surveillance problem

[7, 8, 9]. In [7], the paper considered an approach for monitoring robot performance in a patrolling task and using an auction algorithm to dynamically reassign tasks from those team members that perform poorly. It divides the graph into subsets of nodes and assigns the nodes to each robots. In [10], the paper used a multi-agent Markov decision process (MDP) approach. The authors formulate the planner for each agent as a decentralized MDP and also consider health management of multiple UAV/UGV agents for long-duration missions. Communication constraints are frequently considered in the persistent surveillance or complete coverage problem. Behavior based approaches can also be used [11]. In the paper, they measured performance as time required to generate a complete map of the environment and designed behaviors to enhance global performance.

All these papers did not consider time-interval constraints. It means their environments of the problems have equal priority. However, in [12], in the problem setting, they assumed different priority of the environment. They solved persistent surveillance for tasks with three different priorities. using a vehicle routing problem with time windows.

As for cooperation of heterogeneous teams, the paper of [13] presented decentralized methods for allocating heterogeneous tasks to agents with different capabilities. They used the consensus-based bundle algorithm(CBBA) for the problem. In [14], heterogeneous unmanned ground vehicles(UGV) and micro-air vehicles (MAV) are considered for exploration. The authors applied integer programming for optimization and graph partitioning for dividing agents into multiple teams. There are papers which did not consider heterogeneous agents, but considered different roles of agents [15, 16]. These papers assumed that agents play two roles(maintainers or explorers).

1.2 Contributions

We propose persistent surveillance algorithm with time-interval constraints using mixed integer linear programming in the thesis. The first contribution of the thesis is that we proposed time-interval constraints in persistent surveillance. For the second contribution, the protocol for cooperative movement of heterogeneous agents is considered during surveillance operation.

1.3 Thesis Overviews

The thesis is structured as follows: in chapter 2, we describe setup for the persistent surveillance. Next, we introduce theory of mixed integer linear programming and the problem formulation using MILP in chapter 3. The problem formulation consists of the objective function and constraints(location constraints, movement constraints, capability constraints, and time constraint). In chapter 4, we present the protocol for cooperative movement of heterogeneous agents. Chapter 5 shows the simulation setting and results and conclusion follows.

2

Setup for Persistent Surveillance

In this chapter, we introduce characters of each agent type and then the basic environment setting.

2.1 Heterogeneous Agents

We assume that heterogeneous agents cooperate for persistent surveillance. There are three types of agents. In table 2.1, agents of type A and B are unmanned ground vehicles(UGVs), and an agent of type C is UAV.

Depending on vehicle type, there are different characteristics. First, they have different sensing ranges. A UGV can sense only its current cell. However, a UAV can sense its current located cell and adjacent cells. The moving range is also different. While UGVs can move one cell at a step, UAVs can move two cells at a step. The reason we divide UGVs into two types is that they have different major capabilities. We assume a type A UGV has equipment. For example, in a military scenario, a type A UGV is armed so it is better for defense. On the other hand, type B UGVs are more suitable for surveillance than type A UGVs. Type C UAVs are also suitable for surveillance.

Table 2.1: Characteristics of agents

Type	Vehicle	Sensing range(depth(d))	Moving range(cell/step)	Major capability
Type A	UGV	1	1	Armed
Type B	UGV	1	1	Surveillance
Type C	UAV	2	2	Surveillance

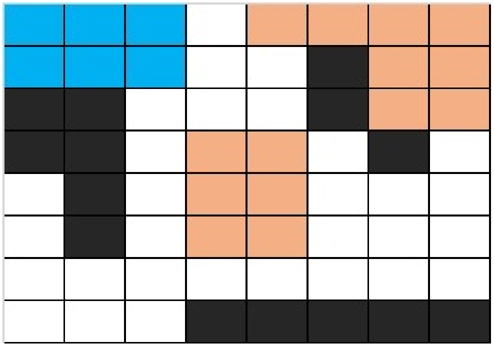
2.2 Environment Setting

The known environment is modeled as a 2-dimensional grid. A cell can be a free cell or an obstacle cell.

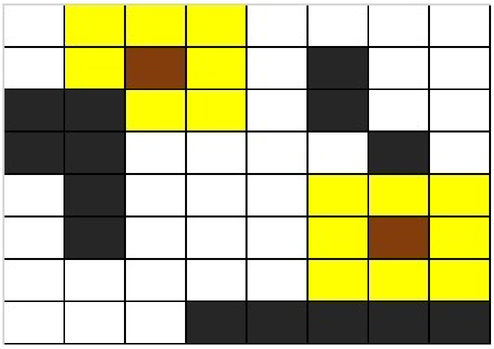
Each cell has its own risk value and time-interval constraint. The risk is changed at each step by agent's movement. The specific rule for changing risk is introduced in chapter 3. As for time-interval constraint, a cell should be visited at least once within specific time-interval by any agent. It is strictly related to priority of the region. If a cell has a small time-interval constraint, it indicates that the cell needs to be observed frequently. Therefore the priority of the cell is high. In opposite, if a cell has a large time-interval constraint, the cell has less priority because it does not need to be observed often.

For example, in Fig. 2.1a, if the orange cells have smaller time interval constraints than the white cells, agents should visit orange cells more often. Conversely, blue cells, which have bigger time interval constraints than white cells, can be visited less often.

In the environment, there are special locations for tasks. Brown cells and yellow cells are special locations in Fig. 2.1b. A brown cell is assumed to be a very dangerous region since it is highly possible for a threat such as a sniper attack. Therefore, it needs protection for surveillance. Only when an agent of type A which is good for defense operation is located in the yellow region and it protects the brown cell against a threat, can an agent of type B or C reconnoitre the brown cell and do a task for three continuous steps. In short, the



(a) the map for time-interval constraints (white, orange, blue : free cell, gray : obstacle cell)



(b) the map for special locations

Figure 2.1: Example of grid environment map

task in special locations should be assigned to teams of heterogeneous agents.

3

MILP Formulations

This chapter describes mixed integer linear program, the objective function for the problem, and important constraints.

3.1 Introduction to MILP

A mixed integer linear program(MILP) is given by **c**, an n-dimensional column vector, **b** an m-dimensional column vector, an m by n matrix **A**, and an n-dimensional column vector of variables, **x**. A MILP is described as

$$
\begin{aligned}
\max \quad & c^T x \\
\textit{subject to} \quad & Ax \leq b, \\
& x \geq 0
\end{aligned}
\tag{3.1}
$$

While all variables are integer in an integer program(IP), if some variables are integer and the others are continuous, that is a MILP. The goal of the problem is to find a vector **x** solving the optimization problem in Eq. 3.1.

MILP has been an interesting approach for decades since it can be applied to various fields such as task assignment, scheduling, and path planning. Also, the optimal solution can be obtained using commercially available software. We solve the problem using the CPLEX optimization software with an AMPL/Matlab interface.

3.2 Objective Function

In chapter 2, we mentioned that each cell has its own risk, and the risk of a cell is changed by agents' movements. An agent moves around to minimize the maximum risk of the region. The objective function at step k can be written as

$$\begin{aligned} \min \{\max_{r^k}\{\, i \in [1, \cdots, N] \,|\, r_i^k\}\} \\ for\, k = 1, \cdots, K \end{aligned} \tag{3.2}$$

In Eq. 3.2, i is a cell index, N is the total number of cells, and r_i^k is the risk of cell i at step k.

3.3 Location Constraints

The first constraint is about the location of agents.

Each agent's location in the region is described by binary variables. x_i^j indicates the binary variable. When an agent j is located in cell i, x_i^j becomes 1. When an agent j is not located in cell i, x_i^j becomes 0.

$$\forall\, j \in [1, \cdots, M]: \qquad \sum_{i=1}^{N} x_i^j = 1. \tag{3.3}$$

where the index of an agent is $j \in [1, \cdots, M]$.

In Eq. 3.3, each agent can occupy only one cell and has to be located in the region, not outside of the region. Therefore, sum of the all binary location variables for each agent becomes 1.

UGV

$$
\begin{aligned}
&\forall\, i \in [1, \cdots, N]: \\
&\quad 0 \;\leq\; \sum_{a=1}^{A} x_i^a + \sum_{b=1}^{B} x_i^b \;\leq 1, \quad when\, cell\, i\, is\, a\, free\, cell \\
&\quad \sum_{a=1}^{A} x_i^a + \sum_{b=1}^{B} x_i^b \;= 0, \quad when\, cell\, i\, is\, an\, obstacle\, cell
\end{aligned}
\tag{3.4}
$$

where the index of a type A agent is $a \in [1, \cdots, A]$ and the index of a type B agent is $b \in [1, \cdots, B]$.

A UGV agent cannot be located in an obstacle cell and also cannot be located in same cell with another UGV agent. To avoidance collision with other UGV agents and obstacles, Eq. 3.4 should be satisfied.

UAV

We assume a UAV agent flies over an obstacle, so constraints about obstacle avoidance are not considered for UAV agents. Eq. 3.5 should be satisfied to avoid collision with other UAV agents.

$$
\begin{aligned}
&\forall\, i \in [1, \cdots, N]: \\
&\quad 0 \;\leq\; \sum_{c=1}^{C} x_i^c \;\leq 1
\end{aligned}
\tag{3.5}
$$

where the index of a type C agent is $c \in [1, \cdots, C]$

3.4 Movement Constraints

Each agent should decide the next position among candidate cells.

$$
\begin{aligned}
&\forall\, j \in [1, \cdots, M]: \\
&\quad \sum_{h=1}^{H} x_{s_h^j}^j \;= 1
\end{aligned}
\tag{3.6}
$$

where the index of a candidate cell is $h \in [1, \cdots, H]$ and s_h^j is a candidate cell of agent j.

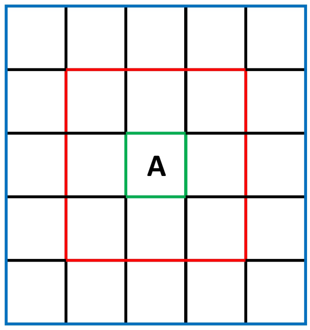


Figure 3.1: Illustration for sensing range and moving range of UGV and UAV

A UGV agent moves one cell per step, and a UAV agent moves two cells per step. In Fig. 3.1, the range of the UGV's candidate cells is inside the red square, and the range of the UAV's candidate cells is inside the blue square. Hence they have different conditions for a candidate cell. A cell becomes a candidate cell if it is satisfies the following conditions.

UGV

- an adjacent cell of the agent
- not occupied by an obstacle

UAV

- an adjacent cell of the agent / a cell which is two cells distant from the agent
- regardless of obstacle occupancy

3.5 Capability Constraints

Sensing ranges are different between UGVs and UAVs. In case of a UGV, it can sense only its current position, one cell, but a UAV can sense its current position and also its adjacent cells. In Fig. 3.1, sensing range of the UGV's candidate cells is inside the green square, and sensing range of the UAV's candidate cells is inside the red square. This fact affects capability of an agent.

In chapter 2, we introduced that each cell has its risk and the objective function is the maximum risk minimization. A risk of a cell is changed at each step by following rules. If an agent is located in cell i,

$$Type\,A: \quad r_i^k = r_i^{k-1} - \alpha \tag{3.7}$$

$$Type\,B: \quad r_i^k = r_i^{k-1} - \beta \tag{3.8}$$

where r_i^k is the risk of cell i at step k, and α and β are risk reduction values of a type A agent and type B agent, respectively.

$$\begin{aligned} Type\,C: \quad r_i^k &= r_i^{k-1} - \gamma \\ r_o^k &= r_o^{k-1} - \gamma \qquad\qquad \forall\, o \in P_i \end{aligned} \tag{3.9}$$

In Eq. 3.9, P_i is the set of cell i's neighbor cell indices and γ is the risk reduction value of a type C agent. Unlike type A and B agents, an agent of type C, which is a UAV, covers up to the right next cells, so it can reduce risk of its current cell and neighbor cells.

If cell i is not covered at step k by any agent, r_i^k is increased by 1 in Eq. 3.10.

$$r_i^k = r_i^{k-1} + 1 \tag{3.10}$$

3.6 Time-interval Constraints

The key point of the proposed algorithm is time-interval constraints. Each cell has its own fixed time-interval constraint and changing time-interval. Like risk reduction capability, time-interval is affected by sensing range. If cell i is covered at step k by a type A or B agent,

$$Type\,A,\ B: \quad T_i^k = 1. \tag{3.11}$$

If cell i is covered at step k by a type C agent,

$$\begin{aligned} Type\,C: \quad & T_i^k = 1 \\ & T_o^k = 1 \qquad \forall o \in P_i \end{aligned} \tag{3.12}$$

However, if no agent covers cell i at step k, the time-interval of cell i at step k is increased by 1 from the time-interval of cell i at step $k-1$ like in Eq. 3.13.

$$T_i^k = T_i^{k-1} + 1 \tag{3.13}$$

To achieve the primary objective of the proposed algorithm, upper bounds of these time-interval values become time-interval constraint values. The constraints can then be described as follows:

$$\begin{aligned} & \forall i \in [1, \cdots, N]: \\ & \qquad T_i^k \leq TC_i \end{aligned} \tag{3.14}$$

where TC_i is the time-interval constraint of cell i.

By this constraint, each cell i can be visited at least once within the specific time interval constraint.

4

Protocol for Cooperative Movement

In chapter 2, we described about special locations for cooperative movement of heterogeneous agents. Now, we first explain the protocol for cooperative movement for special locations in detail from section 4.1 to 4.3, and then the overall algorithm will follow in section 4.4

4.1 Definition of Safe Zone

In the environment, there are special locations that need operation of a heterogeneous team. The location is called a task V_l where l is the task index. In Fig. 2.1b, brown cells are task locations, and yellow cells are defense locations. As we mentioned in chapter 2, only agents of type B and C can be located in task location when an agent of type A is located in defense location. The task is done after the task is covered continuously for three steps. We call agents of type A as a defense team and agents of type B and C as a surveillance team.

For the protocol for cooperative movement, a task V_l defines its safe zone at each step, and uses the safe zone to decide to execute the protocol or not. Radius of the safe zone is

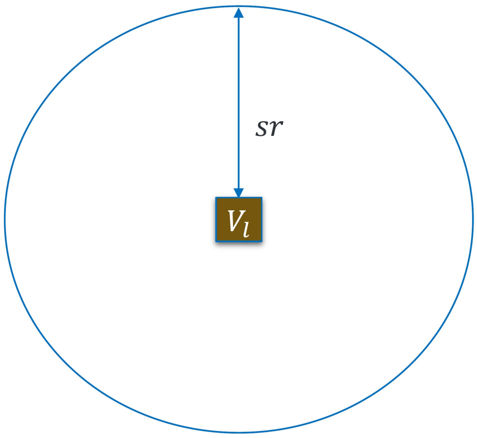


Figure 4.1: Safe zone of the task V_l

described as

$$sr = TC_{V_l} - T_{v_l}^k - \varepsilon \tag{4.1}$$

where sr is the radius of the safe zone and ε is a coefficient to make the safe zone tighter. $TC_{V_l} - T_{v_l}^k$ indicates how many steps remain for keeping the time-interval constraint of task V_l. Namely, the protocol executes when it is possible to break the time-interval constraint. The criteria for execution the protocol is introduced in the next section.

4.2 Task Assignment

Using safe zone of the task V_l at each step, the task decides to execute the protocol or not. Like in Fig. 4.2a, if there is at least one agent from the surveillance team and defense team respectively inside of safe zone, the task V_l is assumed to be safe. But the other cases are not safe situations. Unsafe situations are described as follow:

- The nearest agent(s) of the surveillance team or(and) the defense team is(are) located in boundary of the safe zone.

- The nearest agent(s) of the surveillance team or(and) the defense team is(are) located in outside of the safe zone.

When an unsafe situation of the task V_l occurs, the task V_l is assigned by following steps. First, the task is assigned to the nearest agent of the surveillance team and the nearest agent of the defense team among agents who do not have a task. Second, if there is no agent who does not have a task, the task is just assigned to the nearest one of the surveillance team and the nearest one of the defense team. After two agents are decided to do the task V_l, the agents generate a path from their current position to the task V_l using the A* algorithm. However, in case of the defense agent, it can be located around the task V_l, and cannot be located in the task V_l. So the defense agent uses the path to the right before the cell of the task V_l.

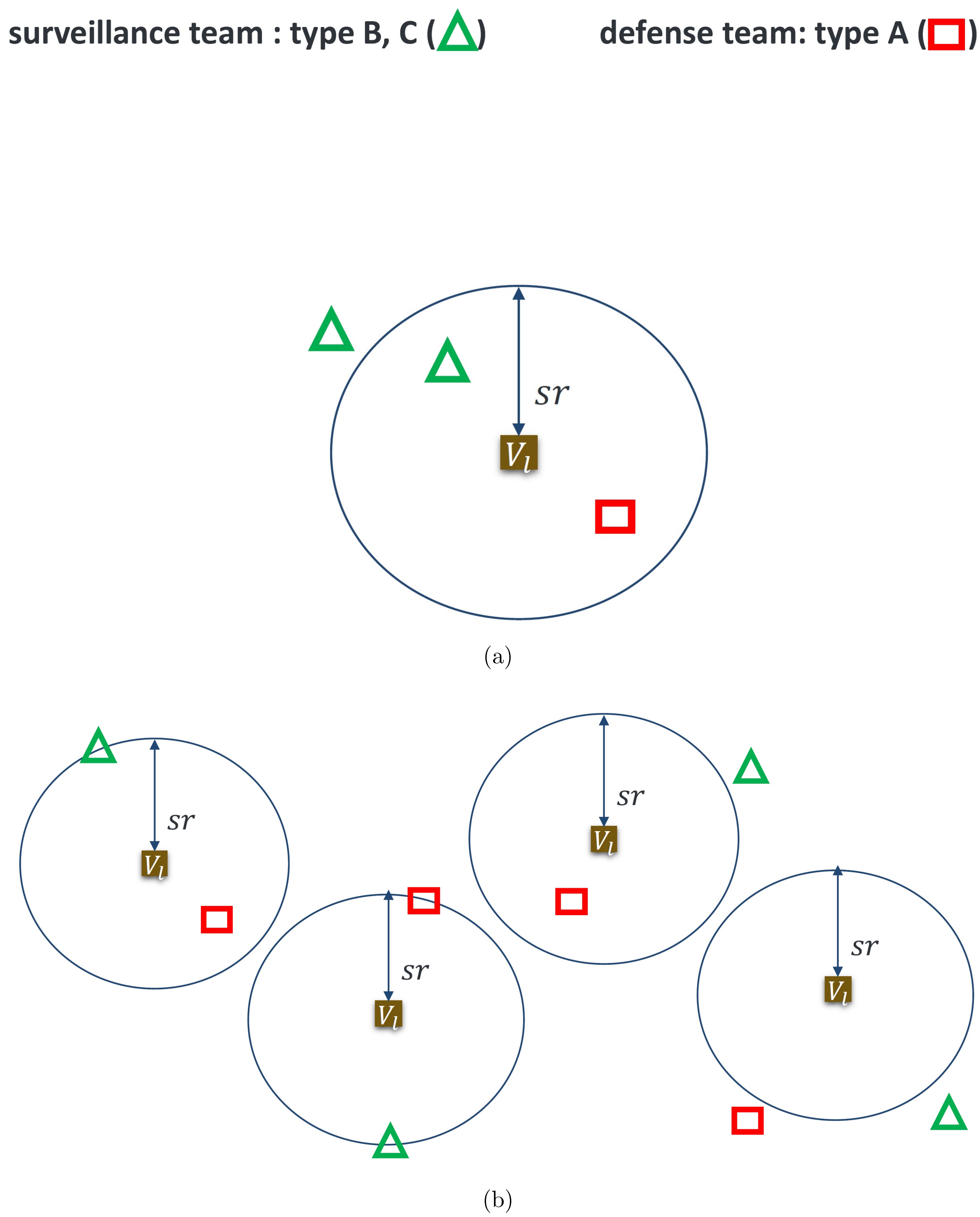


Figure 4.2: Check the status of safety (a)Safe situation (b)Unsafe situations

4.3 Decision Making Behaviour Model

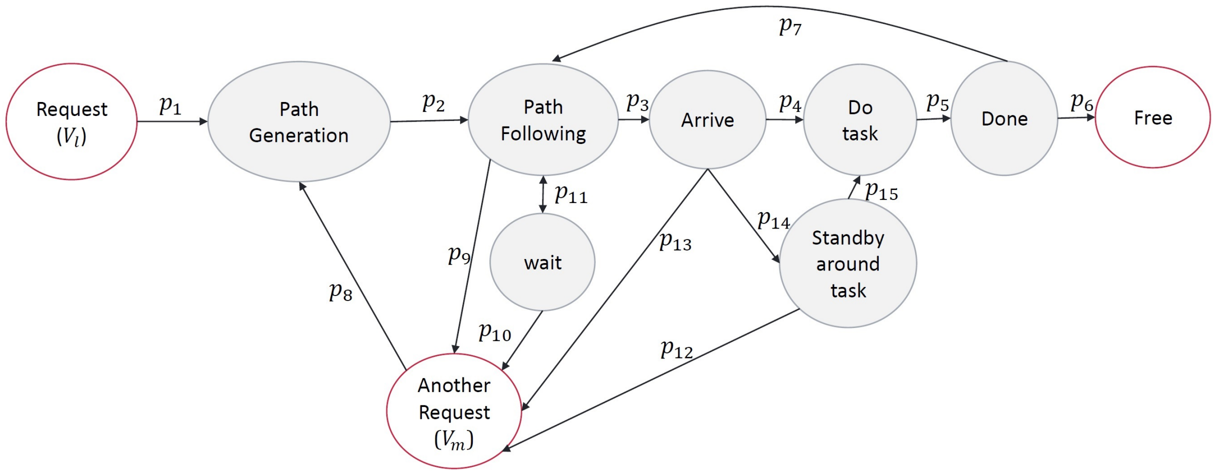


Figure 4.3: Model for decision making behaviour

During the execution of the protocol, various situations could be occurred. In this section, we introduce the decision making behaviour model(Fig. 4.3) to handle these situations. This behaviour model of the agent([17]) is applied to the agent assigned the task, and is not applied to a free agent which is not assigned a task.

The transition relations of the decision making behaviour model are as follows in detail:

- p_1: if the task V_l is assigned
- p_2: if the path is remained
- p_3: if there is one remained the path
- p_4: if the both agents of the surveillance team and the defense team are arrived to the task V_l region

  the surveillance team: arrived to the task V_l

  the defense team : arrived to the defense region(the right next region of the V_l)
- p_5: if the task is done

- p_6: if there is no remained a task
- p_7: if there is remained a task
- p_8: if the agent got the additional task from p_9, p_{10}, p_{12}, and p_{13}
- p_9, p_{10}, p_{12}, and p_{13}: if there is execution of the protocol for additional task and no agent available for the additional task
- p_{11}: if another agent has same location for the next step,

  the defense agent waits until the surveillance agent passes
- p_{14}: if another agent who got same task assignment is not arrived to the task V_l region
- p_{15}: if all agents who got same task assignment are arrived to the task V_l region

4.4 Overall Algorithm

In Fig. 4.4, we introduce the overall algorithm of the thesis. When the program starts, MILP with general constraints, as explained in chapter 3 runs using initial risk, time-interval, and agents' locations. From MILP, we get the results which are risk, time-interval, and agents' locations. From this information, it decides execution of the protocol for cooperative movement or not. If an agent gets the task, the agent runs MILP with specific path constraints for the task. If an agent gets no task, the agent runs MILP with general constraints as before. This procedure iterates continually.

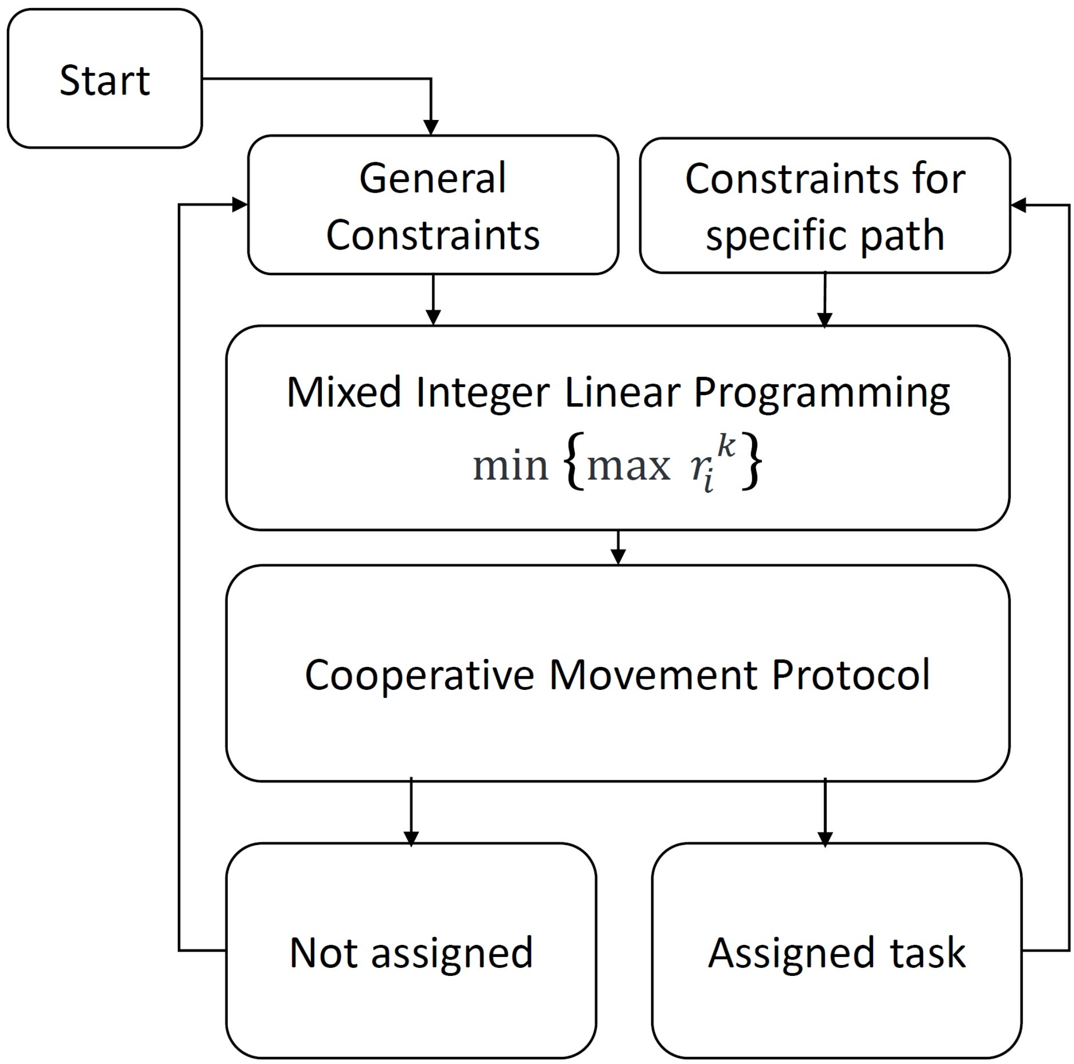


Figure 4.4: Overall algorithm

5

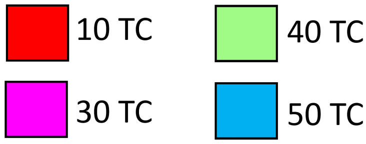
Simulation

5.1 Simulation Setting

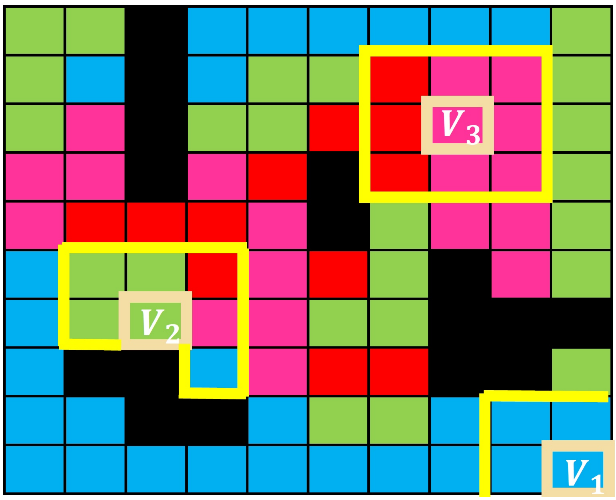
For simulation, we assume a 10x10 grid environment and two agents of type A, type B, and type C, respectively. It means there are four members of the surveillance team and two members of the defense team. The obstacle ratio is 0.15 and black cells are an obstacle cells in Fig. 5.1a. There are three special locations for cooperative operation of heterogeneous agents. In Fig. 5.1b, the tasks V_1, V_2, and V_3 are located in special locations and yellow regions around the tasks are defense regions. As you see, we assume more tasks than the number of defense agents, so it is possible for to get occur one agent to get two tasks at the same time step.

As for time-interval constraints, the entire region consists of cells with different time interval constraints in Fig. 5.1a and 5.1b. Obviously, if a cell has the smallest time-interval constraint, the cell is the most important cell to observe. Conversely, if a cell has the largest time-interval constraint, the cell is the lowest priority within the entire region.

The risk of each cell is initialized to 100 at first, and risk reduction capability differs by



(a) Time-interval constraint of a cell



(b) Grid environment map for simulation

Figure 5.1: Simulation environment

type of agents. A type A agent can reduce the risk by 20, and the risk reduction capability of a type B agent is 25. Also, a type C agent can reduce risk by 23. If more agents than one agent cover a cell, the risk of the cell is reduced by the sum of each agent's risk reduction capability. We also assume a lower bound of risk as 0, and high risks pop up at unexpected time step.

5.2 Simulation Results

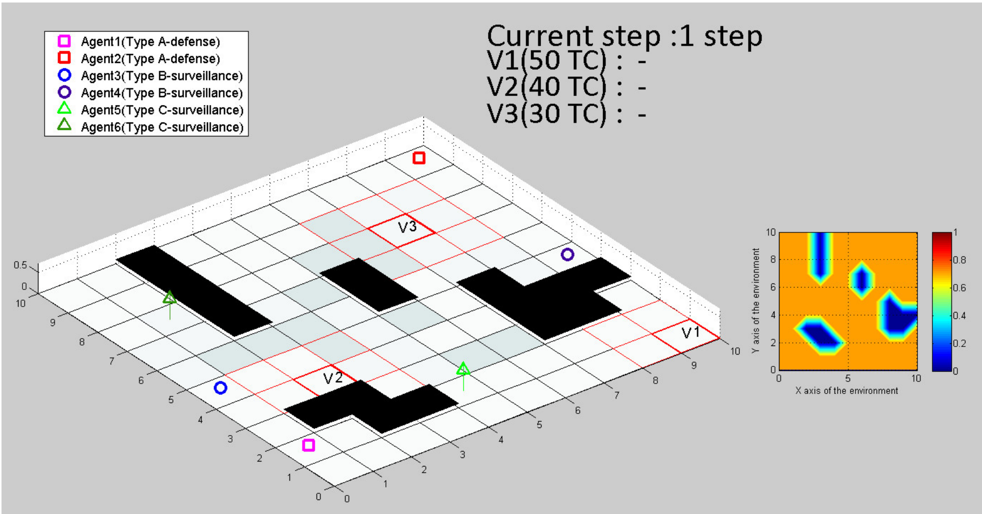
Simulation video images are described from Fig. 5.2 to Fig. 5.5. We would like to explain the simulation video first. In the simulation video, the left bottom figure is the entire environment. Squares tinged with red are type A UGVs, and circles tinged with blue are type B UGVs. Triangles with green are type C UAVs. When a task V_l executes, a red bold line appears, and the team of heterogeneous agents assigned the task V_l change their shape to be bigger and bolder. The center top message is the announcement for the current step and execution of the protocol for cooperative movement. If the protocol of a task executes, a CALL massage appears. Also, the right figure is the risk distribution for the entire environment at each step.

There are moments when all three tasks are executed at the same step. As we mentioned, because the number of defense agents is two and the number of tasks is three, one defense agent should do two tasks in this situation. The situation is described from Fig. 5.3 to Fig. 5.5.

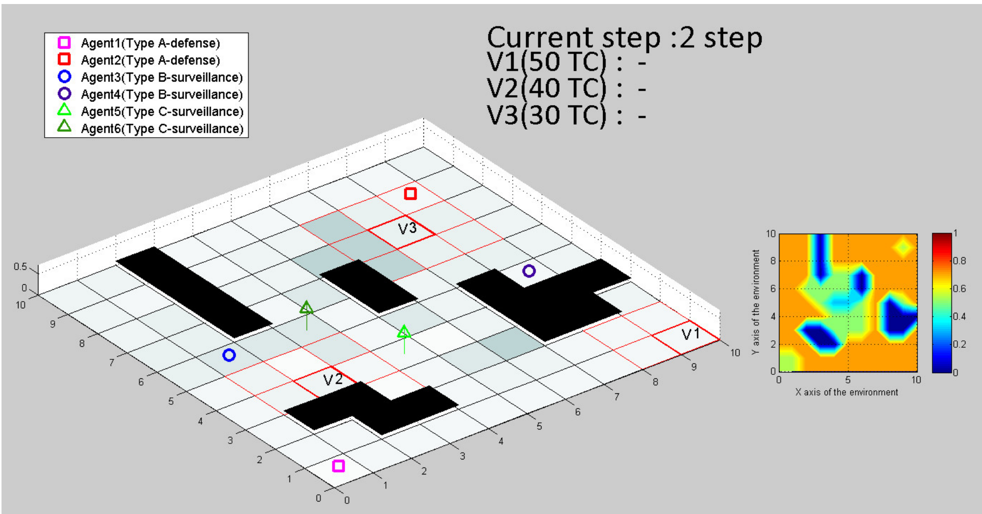
The objective function of the proposed algorithm is the maximum risk minimization. In Fig. 5.6, we show the maximum risk is decreased eventually to about 30 in plus-minus 15 range. Even though high risk unexpectedly occurred, the maximum risk is reduced within the maximum 4 steps. Since cells have different time-interval constraints, risk reduction speed and convergence range of each time-interval constraint are various. In case of a cell with $10TC$, agents covered the cell most often so speed of risk reduction is the fastest, and the convergence range of risk is the lowest.

Fig. 5.7 shows the state of safety for each task V_l. 1 indicates that the task region is safe, so there is no need to execute the protocol for cooperative movement. In the other cases, it is 0. With the results, each task calls a team of heterogeneous agents once within its time-interval constraint.

As for time-interval constraints, all time-interval constraints are well satisfied in Fig. 5.8. Each graph is a histogram of the maximum time-interval at each step amongst cells with the same time-interval constraint. For example, in case of first histogram, we found the cells which have 10 time-interval constraint only, and then we choose the maximum time-interval among these cells at each step. Therefore, the histogram results show all cells covered within its time-interval constraint perfectly.

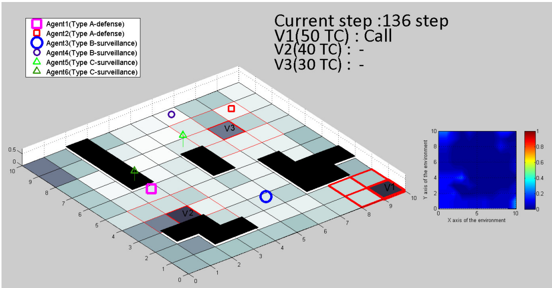


(a) Step 1

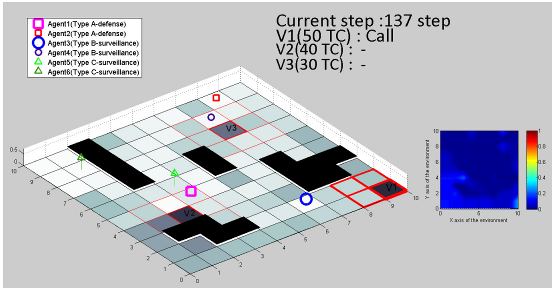


(b) Step 2

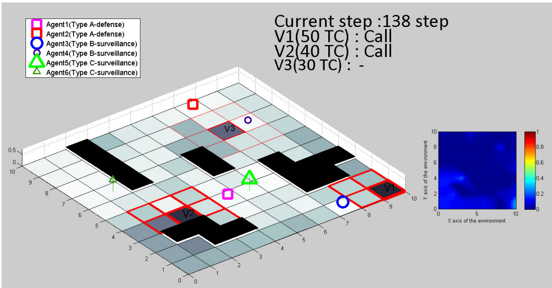
Figure 5.2: Simulation video image: start



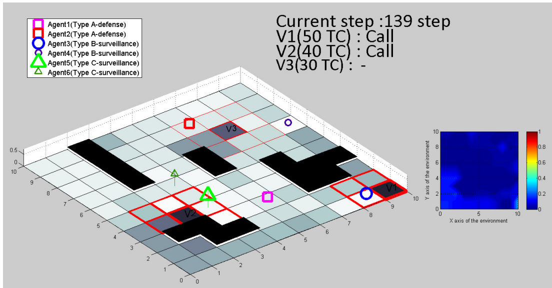
(a) Step 136



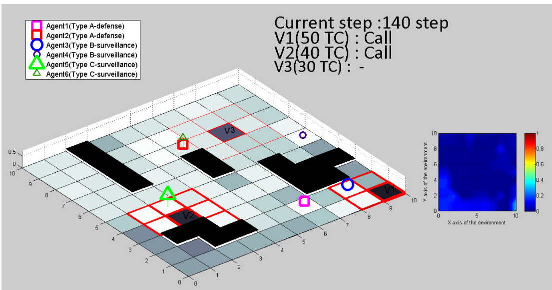
(b) Step 137



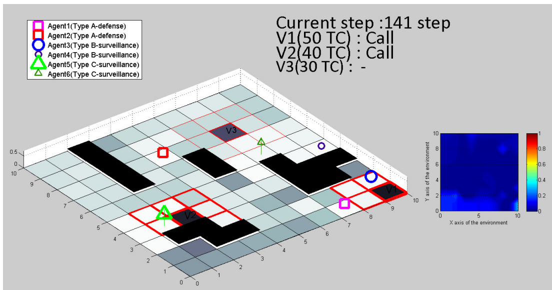
(c) Step 138



(d) Step 139

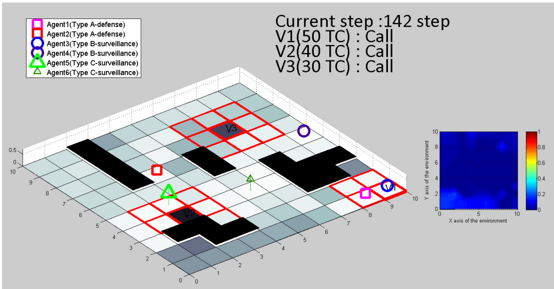


(e) Step 140

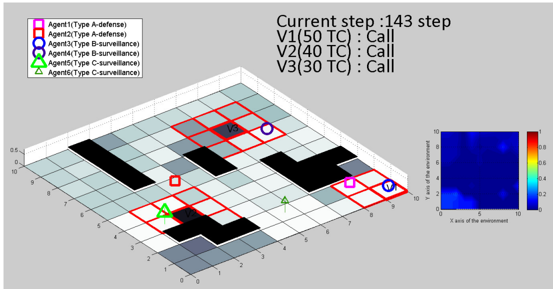


(f) Step 141

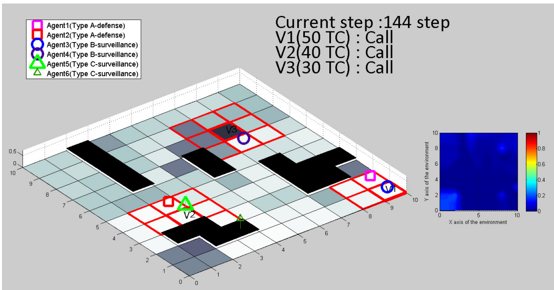
Figure 5.3: Simulation video image: the protocol for cooperative movement(from step 136 to step 141)



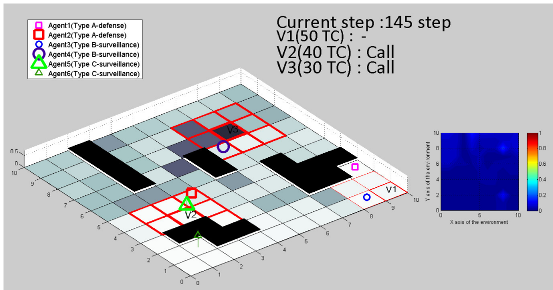
(a) Step 142



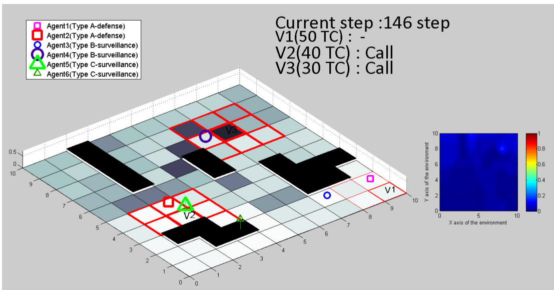
(b) Step 143



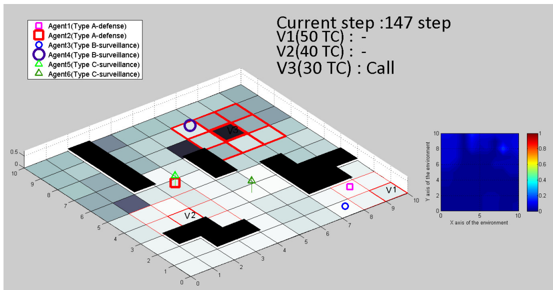
(c) Step 144



(d) Step 145

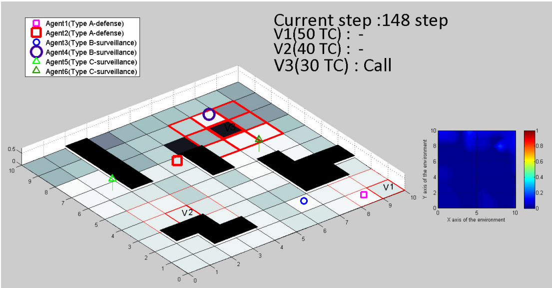


(e) Step 146

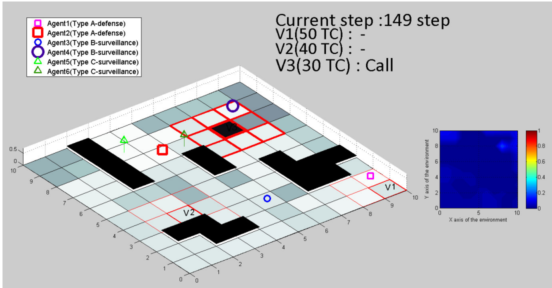


(f) Step 147

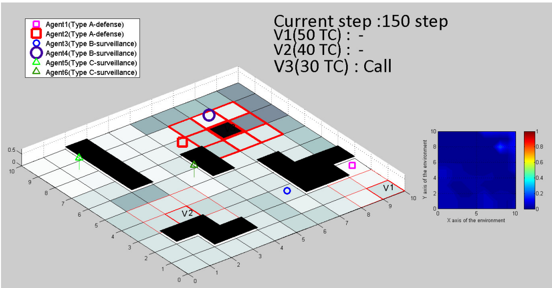
Figure 5.4: Simulation video image: the protocol for cooperative movement(from step 142 to step 147)



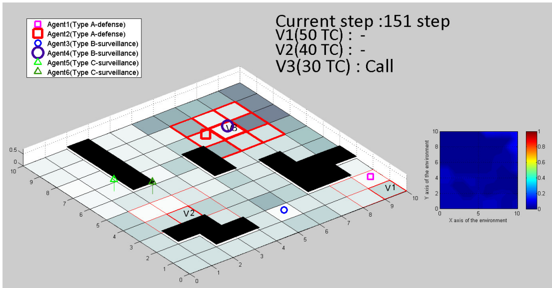
(a) Step 148



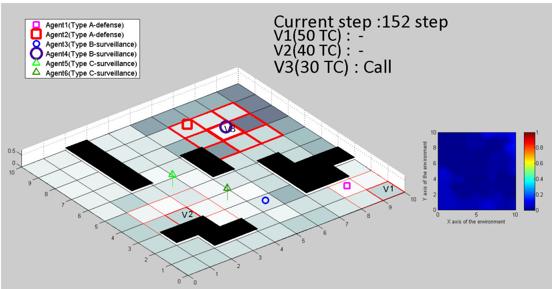
(b) Step 149



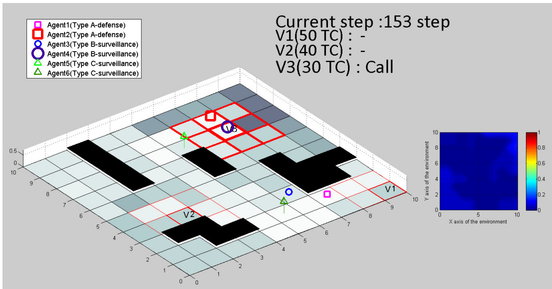
(c) Step 150



(d) Step 151

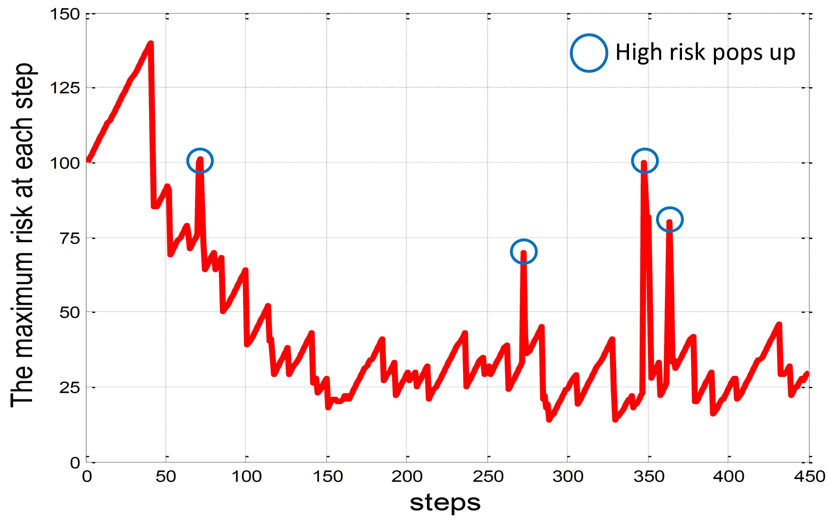


(e) Step 152

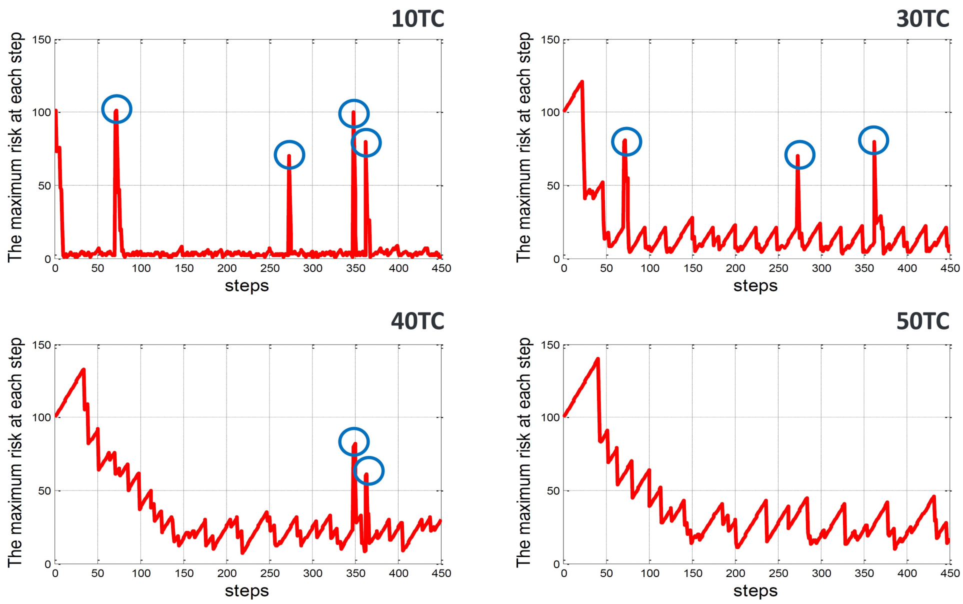


(f) Step 153

Figure 5.5: Simulation video image: the protocol for cooperative movement(from step 148 to step 153)



(a)



(b)

Figure 5.6: Performance measure(a)The maximum risk at each step (b)The maximum risk at each step amongst cells with the same time-interval constraint

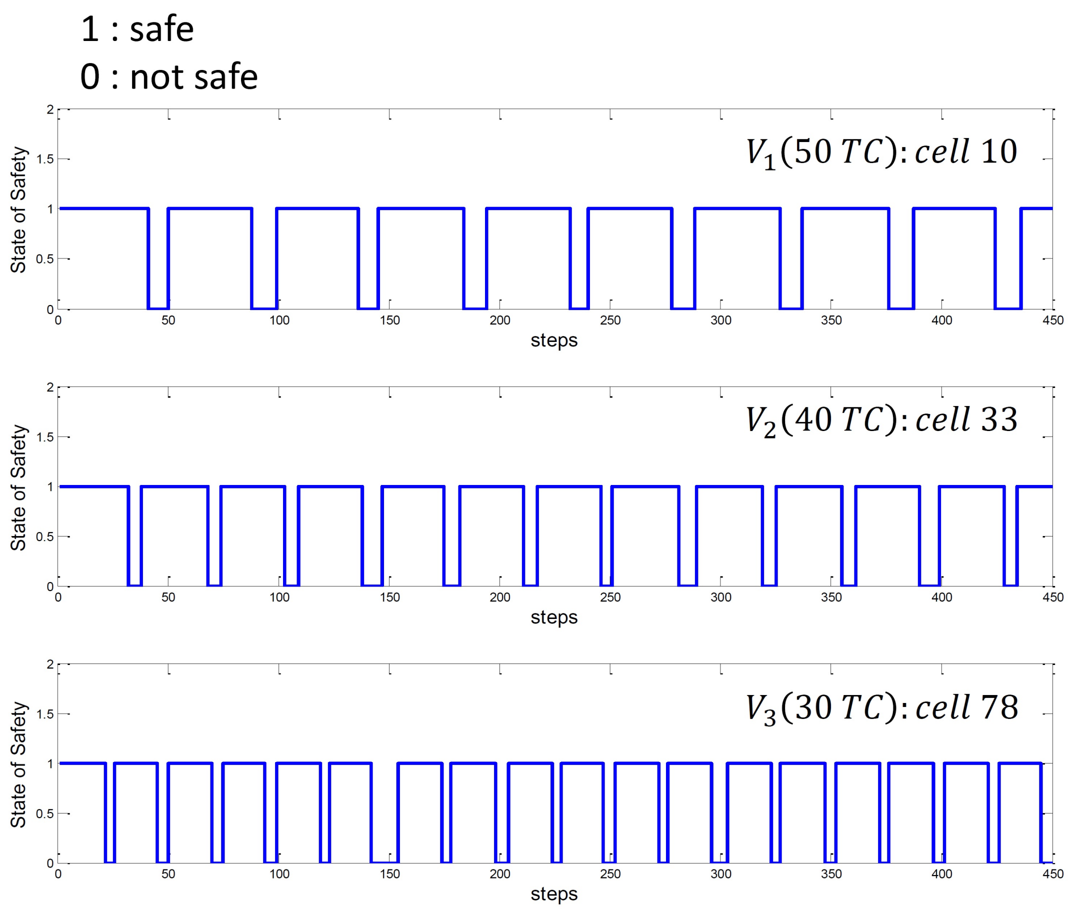


Figure 5.7: State of safety for each task

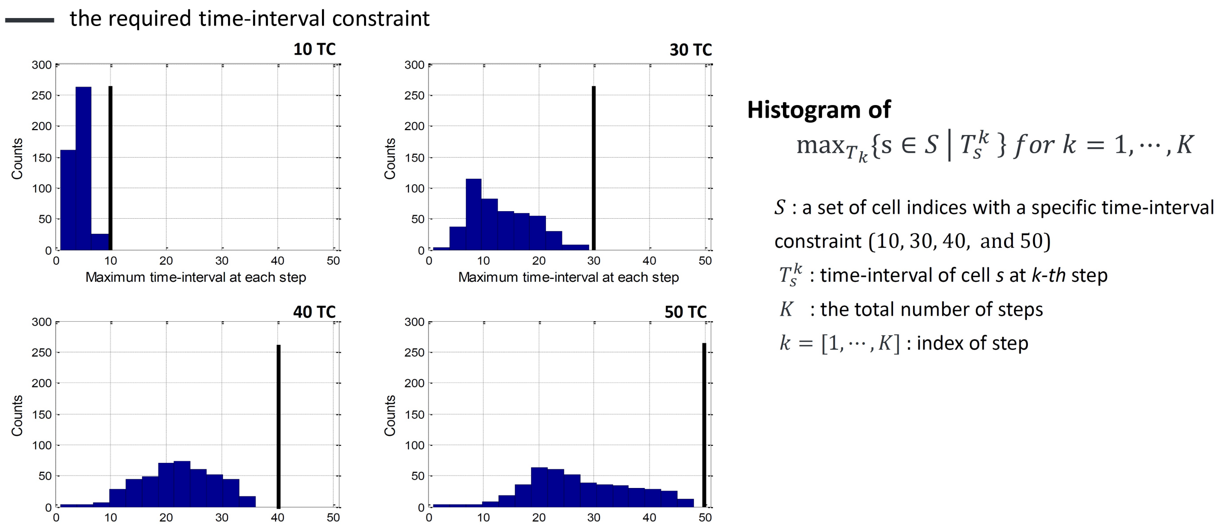


Figure 5.8: Histogram of the maximum time-interval at each step amongst cells with the same time-interval constraint

6 Conclusions

In this thesis, we proposed a multi-agent centralized path planning algorithm for persistent surveillance using MILP. The key point of the thesis is time-interval constraints are involved in the problem. Also, we consider three types of agent and the protocol for cooperative movement of team of heterogeneous agents.

By using MILP, we got local optimal results for the maximum risk minimization subject to time-interval constraints. In the simulation result, the maximum risk of the entire region is decreased to certain range and agents can handle unexpected high risk. Also, The simulation results show agents explore the region within all constraints and cooperate with other heterogeneous agents for special tasks efficiently.

References

[1] Yann Chevaleyre, Francois Sempe, and Geber Ramalho. A theoretical analysis of multi-agent patrolling strategies. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 3*, pages 1524–1525. IEEE Computer Society, 2004.

[2] Brett Bethke, Joshua Redding, Jonathan P How, Matthew A Vavrina, and John Vian. Agent capability in persistent mission planning using approximate dynamic programming. In *American Control Conference (ACC)*, pages 1623–1628. IEEE, 2010.

[3] Eduardo Arvelo, Eric Kim, and Nuno C Martins. Maximal persistent surveillance under safety constraints. In *Robotics and Automation (ICRA), IEEE International Conference on*, pages 4048–4053. IEEE, 2013.

[4] Nikhil Nigam and Ilan Kroo. Persistent surveillance using multiple unmanned air vehicles. In *Aerospace Conference, 2008 IEEE*, pages 1–14. IEEE, 2008.

[5] Anshika Pal, Ritu Tiwari, and Anupam Shukla. Communication constraints multi-agent territory exploration task. *Applied Intelligence*, pages 1–27, 2013.

[6] Fabio Pasqualetti, Antonio Franchi, and Francesco Bullo. On cooperative patrolling: optimal trajectories, complexity analysis, and approximation algorithms. *Robotics, IEEE Transactions on*, 28(3):592–606, 2012.

[7] Charles Pippin, Henrik Christensen, and Lora Weiss. Performance based task assignment in multi-robot patrolling. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, pages 70–76. ACM, 2013.

[8] Weihua Sheng, Qingyan Yang, Jindong Tan, and Ning Xi. Distributed multi-robot coordination in area exploration. *Robotics and Autonomous Systems*, 54(12):945–955, 2006.

[9] Robert Zlot, Anthony Stentz, M Bernardine Dias, and Scott Thayer. Multi-robot exploration controlled by a market economy. In *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, volume 3, pages 3016–3023. IEEE, 2002.

[10] Bullet Josh Redding, Tuna Toksoz, N Kemal Ure, Alborz Geramifard, Jonathan P How, Matthew A Vavrina, and John Vian. Distributed multi-agent persistent surveillance and tracking with health management. In *AIAA Guidance, Navigation, and Control Conference (GNC)*, 2011.

[11] Jose Vazquez and Chris Malcolm. Distributed multirobot exploration maintaining a mobile network. In *Intelligent Systems, Proceedings. 2004 2nd International IEEE Conference*, volume 3, pages 113–118. IEEE, 2004.

[12] Ethan Stump and Nathan Michael. Multi-robot persistent surveillance planning as a vehicle routing problem. In *Automation Science and Engineering (CASE), 2011 IEEE Conference on*, pages 569–575. IEEE, 2011.

[13] Han-Lim Choi, Andrew K Whitten, and Jonathan P How. Decentralized task allocation for heterogeneous teams with cooperation constraints. In *American Control Conference (ACC)*, pages 3057–3062. IEEE, 2010.

[14] Ayush Dewan, Aravindh Mahendran, Nikhil Soni, and Madhava Krishna. Optimization based coordinated ugv-mav exploration for 2d augmented mapping. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pages 1125–1126. International Foundation for Autonomous Agents and Multiagent Systems, 2013.

[15] Yuanteng Pei and Matt W Mutka. Steiner traveler: Relay deployment for remote sensing in heterogeneous multi-robot exploration. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 1551–1556. IEEE, 2012.

[16] Piyoosh Mukhija, K Madhava Krishna, and Vamshi Krishna. A two phase recursive tree propagation based multi-robotic exploration framework with fixed base station constraint. In *Intelligent Robots and Systems (IROS), IEEE/RSJ International Conference on*, pages 4806–4811. IEEE, 2010.

[17] S Jeyaraman, A Tsourdos, R Zbikowski, and BA White. Kripke modelling of multiple robots with decentralized cooperation specified with temporal logic. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 219(1):15–31, 2005.

국 문 초 록

본 논문에서는 각 지역을 방문하는 시간 간격이 구속조건으로 주어져 있는 상황에서 다중 로봇 시스템이 지속적인 정찰 임무를 수행하는 문제를 정식화 하고, 혼합정수계획법을 사용하여 접근하였다. 정찰 임무에 대한 대부분의 기존 연구들이 지역의 중요도를 달리하여 고려하지 않은 반면, 본 논문에서는 전체 환경에서 각 지역이 다른 중요도를 갖고 있는 경우를 고려하고, 이를 최소 한번 이상 정찰이 되어야 하는 시간 간격 구속조건과 함께 반영하여 연구를 진행하였다. 또한 기본적인 정찰 임무와 함께, 이종(異種) 로봇들의 협업을 필요로 하는 특이 지역이 있는 경우 이러한 임무를 수행하기 위한 협업 프로토콜을 설계하였다. 경로 계획 부분은 혼합정수계획법을 이용하였고, 이에 특정 지역의 임무 할당을 위한 협업 프로토콜 알고리즘을 통합 구성하였다. 환경을 지속적으로 탐색/정찰하는 임무에 대한 시뮬레이션을 수행한 결과, 정찰 임무가 지속 될수록 지역 내의 최대 위험도 값이 감소하여 일정 범위 내에서 수렴하고, 혼합 정수계획법을 위한 구속조건들이 만족되었다. 또한 지역 내의 모든 셀들이 정해진 시간 간격 내에 최소 한번 이상 방문되는 것과 이종 로봇들의 협업이 필요한 특이 지역 또한 해당 시간 간격 제한 안에 정확히 한번 프로토콜이 활성화 되어 로봇 팀에 의해 감지되는 것을 확인하였다.

주요어: 혼합 정수계획법, 지속 정찰, 경로 계획, 다중 로봇, 협업
학 번: 2012-20712