Multi-Robot Fence Patrol in Adversarial Domains

Noa Agmon, Sarit Kraus and Gal A Kaminka [1]
Department of Computer Science
Bar Ilan University, Israel
{*segaln,sarit,galk*}*@cs.biu.ac.il*

Abstract. This paper considers the problem of multi-robot patrolling along an open polyline, for example a fence, in the presence of an adversary trying to penetrate through the fence. In this case, the robots' task is to maximize the probability of detecting penetrations. Previous work concerning multi-robot patrol in adversarial environments considered closed polygons. That situation is simpler to evaluate due to its symmetric nature. In contrast, if the robots patrol back and forth along a fence, then the frequency of their visits along the line is coherently non-uniform, making it easier to be exploited by an adversary. Moreover, previous work assumed perfect sensorial capabilities of the robots in the sense that if the adversary is in the sensorial range of the robot is will surely be detected. In this paper we address these two challenges. We first suggest a polynomial time algorithm for finding the probability of penetration detection in each point along the fence. We then show that by a small adjustment this algorithm can deal with the more realistic scenario, in which the robots have imperfect sensorial capabilities. Last, we demonstrate how the probability of penetration detection can be used as base for finding optimal patrol algorithms for the robots in both strong and weak adversarial environment.

Keywords. Multi-Robot systems, Adversarial/game domains, Multi-robot path planning

1. Introduction

The problem of multi-robot patrol has been investigated during the past few years [3,4,8]. The subject is of interest for a number of reasons, mainly its applicability in various domains, for example floor cleaning or lawn mowing [6]. In this problem, a team of robots is required to continuously visit the area, borders of a closed area or an open polyline, while monitoring it in order to detect change in state. Most of the studies in this area concentrated in assuring optimization of frequency criteria [4, 8]. However, several studies considered the problem of multi-robot patrol in adversarial environment, in which the robots are required to patrol along a given area in order to maximize the probability of detecting an adversary which attempts to penetrate through the patrol while trying to evade the robots. This problem is applicable in many security applications [1, 2, 11].

[1]This research was partially funded by ISF Grant #1357/07.

In this work we concentrate on multi-robot patrol in adversarial settings *along an open polyline*, for example a fence. In such environments, the k robots are required to travel back and forth along the line in order to maximize the probability of detecting penetrations.

Recent studies of multi-robot patrol in adversarial environment concentrated on patrolling around closed areas, i.e, a closed polyline (circle) [1]. The open polyline environment is more complicated to analyze, due to several reasons. The main reason lies in the fact that when a robot travels back and forth along a line, the frequency is inherently non-uniform. Therefore in adversarial environments the adversary can take advantage of this fact and manage to penetrate more easily. This forces the robots to adopt an asymmetric behavior in the sense that at each point along the line the best action of the robot might be different.

Another assumption made in previous work concerns the perfect sensorial capabilities of the robots with respect to detecting the penetration. In [1, 2] the authors assume that the robots detect penetrations with probability 1 once the adversary lies within the sensorial range of a robot. However, in reality this might not be the case. In many cases, the system can guarantee that the robot will detect the adversary up to some percentage, i.e., with probability p_d.

In this paper we tackle these two challenges. We first contrast the two environments - patrol along closed and open polylines, and show the consequence of the difference between the two with respect to the calculation of probability of penetration detection along the polyline.

We provide a polynomial time algorithm for finding the probability of penetration detection (ppd) in each segment along the open polyline, with respect to the location and direction of the robots. We then consider robot with imperfect detection, i.e., $p_d < 1$, and describe an algorithm that finds the probability of penetration detection in each segment with respect to this p_d value.

Finally, we show that after establishing the probabilities of penetration detection along the line, this information can be used as base for finding the *optimal* patrol algorithm in different adversarial environments. We demonstrate how the ppd functions are used in order to find the optimal patrol algorithm in a strong adversarial model, in which the adversary has full knowledge of the patrol scheme. We then show that if the robots are faced against a weak adversary that chooses its penetration spot at random, the optimal patrol that maximizes the expected ppd is no longer the deterministic algorithm as in the case of a closed polyline.

2. Related work

Systems of multiple robots engaged together in order to patrol in adversarial environments has been studied in various contexts. The problem is related to the multi-robot coverage (e.g. [5]), in which a team of robots is required to visit a target area *once*. However, in the area patrol problem (e.g. [3, 8]) and the fence patrol problem [9], the area is visited *continuously* by the team of robots while usually attempting to maintain some frequency criteria.

Agmon et. al. [1] introduced the multi-robot perimeter patrol in adversarial environments, along with the robotic model we base our work upon. In their work, they make worst-case assumptions regarding the capabilities of the robots, i.e., they assume the adversary has full knowledge of the robots' patrol algorithm.

They provide a polynomial time algorithm for finding the probability of penetration detection throughout the perimeter, and describe a second algorithm for using these probability functions in order to find the patrol algorithm that maximizes the minimal probability of penetration detection along the perimeter. As mentioned previously, they assume that the detection of the adversary is perfect.

Agmon et. al. [2] further study the perimeter patrol problem in different adversarial environments. They study the case in which the adversary has no knowledge of the patrol algorithm, and chooses its penetration spot at random with uniform distribution, and show that the simple deterministic algorithm maximizes the expected `ppd` in this case. They examine the compatibility of different algorithms to different adversarial settings also empirically, and have shown that the algorithm intended for the worst case scenario failed miserably when working against a random adversary, and that the deterministic algorithm failed against a strong adversary. Also in this work Agmon et. al. do not refer to imperfect sensorial capabilities of the robots.

Other closely related work is the work by Paruchuri et. al. [11, 12], which consider the problem of placing security checkpoints in adversarial environments. They use policy randomization for the agents behavior in order to maximize their rewards. In their work, the adversary has full knowledge of the agents, therefore it uses it in order to minimize its probability of being caught in some checkpoint.

Elmaliach et. al [9] consider the fence patrol problem, however they aim to maximize optimization criteria using a deterministic algorithm, and do not assume the existence of an adversary. They provide a mathematical model that takes into consideration drawbacks of the system, for example odometry problems, when trying to find a patrol algorithm that maximizes the optimization criteria. Specifically, they show that overlapping the sections of responsibility given to each robot can be advantageous, and the optimality of the patrol algorithm is characterized by the extent of the overlap. In this paper, we show that in an open fence the deterministic algorithm is no longer optimal for both adversarial model, hence using this algorithm is not beneficial.

Theoretical work based on stochastic processes that is related to our work is the *cat and mouse* problem [7], also known as the *predator-prey* [10] or *pursuit evasion* [13]. In this problem, a cat is attempting to catch a mouse in a graph where both are mobile. The cat has no knowledge about the mouse's movement, therefore as far as the cat is concerned, the mouse travels similarly to a simple random walk on the graph. We, on the other hand, have different assumptions about the adversary, consider a *robotic* model, in which the movement is correlated to the movement of a robot, and in our model the robots travel along a fence, rather than in a graph or an area.

3. Background

3.1. Robotic and environmental model

In our work, we assume to be given a team of k homogenous robots that are required to patrol along an open polyline. The line is divided into N segments (not necessarily of the same length or orientation - see figure 1c.), where each robot travels through one segment per one time cycle. This allows us to consider heterogenous domains, i.e., domains in which the velocity constraints along the

perimeter can change. Note that the polyline is not necessarily straight, but can be of any open shape. We then divide the N segments into k sections, where each section contains $d = N/k$ segments. We therefore focus our analysis of the system to analysis of a section, since the behavior in equivalent in all sections.

We assume the robots have directionality associated with their movement. In each time cycle, the robots have to decide where they should go. Since this is a line, they have the option to continue in their current course or turn around. Turning around may be costly, hence we model this cost by the number of time cycles it takes the system to stabilize after all robots turn around, denoted by τ.

The patrol algorithm of the robots is characterized by the probability p, i.e., the robots go straight forward with probability p, and turn around with probability $q = 1 - p$.

In our adversarial models, the adversary had to decide at time 0 through which segment to penetrate. We assume that the time it takes it to penetrate is not instantaneous, and lasts t time units.

3.2. Patrolling along a closed polyline vs. an open polyline

As stated previously, recent studies in multi-robot patrol in adversarial environments [1,2] assume the robots travel around a closed, circular, area. In the following, we describe why patrolling along an open polyline has a challenging nature compared to patrolling in cyclic environments (closed polyline).

The first reason lies in the fact that the robots are required to go back and forth along a part (or parts) of the open polyline, therefore the elapsed time between two visits of a robot in each point along this line can be almost twice as long as the elapsed time in a circular setting. In Figure 1, we are given two environments: one closed polyline (circle) (a) and an open polyline (b). Note that the open polylines b. and c. are equivalent in the sense that each robot travels through one segment per time step, regardless of the shape of the section. Both lines a. and b. are of the same total length l and with the same number of robots (4). In the circular environment, if it takes an adversary more than $l/4$ time units to penetrate - it will never be able to penetrate even if the robots simply travel with uniform distance between them continuously. However, if the robots travel along an open polyline (b), the maximal time duration between two visits of the robot — even in the best case, is $2l/4 - 2$ [9]. Therefore a weaker adversary that has penetration time which is almost twice as long as in the circular fence might still be able to penetrate.

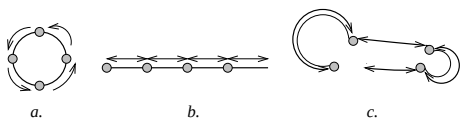

Figure 1. Illustration of the difference between patrolling along a line and patrolling along a circle, for different polylines

Another reason for the added complication in analyzing the probability of penetration detection in open polyline environments lies in the asymmetric nature of traveling in the segments along time. In a circular environment, if the robots

are coordinated and switch directions in unison, then the placement of the robots is symmetric in each time unit. Therefore all segments in the same distance from some robot (with respect to its direction) have the same probability of penetration detection. Hence in order to calculate the optimal way of movement (in our case the probability p of turning around), it is enough to consider only one section of d segments, and the resulted p is equivalent throughout the execution [1]. In an open polyline environment this is not the case. The probability of penetration detection differs with respect to the current location and direction of the robot. Therefore the algorithm that finds the ppd for each segment, needs to calculate the ppd as a function of p for each segment s_i for each possible initial location of the robot inside the section. Therefore it results with a matrix of size $d \times d$ of the ppd functions (as opposed to a vector of d functions in the circular fence).

4. Patrol along an open polyline

The *best* patrol algorithm for a team of robots in adversarial environment requires to find the algorithm that maximizes some function of the probability of penetration detection (ppd). This function depends on the assumptions made on the capabilities of the adversary [2]. However, the basic step upon which other calculations are made is finding the probability of penetration detection in all segments along the fence. Therefore in this section we describe Procedure FindFencePPD that finds the ppd in the segments.

The ppd in a segment s_i is determined by the probability of the *first* arrival to s_i during t time units. This is due to the fact that we assume, at this stage, that if the robot arrives at a segment that is currently occupied by the adversary, it will detect it, hence it does not matter if the segment is visited more than once during t time units. For simplicity, we discuss the case in which $\tau = 1$.

Denote the ppd in segment s_i by ppd_i. We describe the system as a Markov chain (see Figure 2). Since the robots have directionality associated with their movement, we create two states for each segment: the first for traveling in a segment in the clockwise direction, and the second for traveling in the counterclockwise direction. The probability of turning around at the end of each section is 1, otherwise the robot is going straight with probability p, and turns around with probability $q = 1 - p$.

The algorithm is dynamic programming inspired, and uses the state transition rules describes in Figure 2 in order to fill in a matrix M gradually, until reaching the state of the system after t time units (last line). The algorithm works as follows. Each segment s_i is associated with a "phantom" variable f_i. Whenever an element goes through s_i's location in the matrix, it is multiplied by f_i. At the end, all visits to the segment are added, and all fantom variables are substituted with 1 *except* for f_i. Therefore the polynomial coefficients of f_i^t is the probability of the t'th visit to s_i, the coefficient of f_i^{t-1} represents the probability of the $t-1$ visit, and so on. Hence ppd_i in this case is the polynomial visit of f_i^1, i.e., the probability of first visit to s_i.

Recall that $d = N/k$. The returned value from Procedure FindFencePPD is a matrix of size $d \times d$, where each row i contains d functions representing the ppd in each segment, given that the robot is currently in segment i.

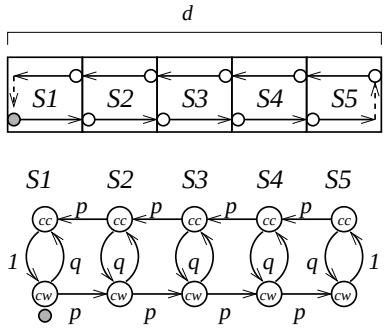

Figure 2. Description of the system as a Markov chain, as base for the FindFencePPD algorithm.

Algorithm 1 Procedure FindFencePPD(d, t)

for $loc \leftarrow 1$ to d **do**
  Create the matrix Res of size $d \times d$, initialized with 0s.
  Create the matrix M of size $2d \times (t+1)$ initialized with 0s.
  Set $M[1, s_{loc}^{cw}] \leftarrow f_{loc}$
  Fill all entries in M gradually using the following rules.
  for $r \leftarrow 2$ to t **do**
    Set $M[r, s_1^{cw}] \leftarrow f_1 \times M[r-1, s_1^{cc}]$
    Set $M[r, s_d^{cc}] \leftarrow f_1 \times M[r-1, s_d^{cw}]$
    for all other entries in row r **do**
      Set $M[r, s_i^{cw}] \leftarrow v_i \times \{p \cdot M[r-1, s_{i+1}^{cw}] + q \cdot M[r-1, s_i^{cc}]\}$.
      Set $M[r, s_i^{cc}] \leftarrow v_i \times \{p \cdot M[r-1, s_{i-1}^{cc}] + q \cdot M[r-1, s_i^{cw}]\}$.
  Create the vector V of size d
  for $j \leftarrow 1$ to d **do**
    $V[j] \leftarrow \sum_{i=1}^{t} M[i, s_j^{cw}] + M[i, s_j^{cc}]$
    For each entry i in V, substitute all f_k, $k \neq j$ with 1.
    Set $Res[loc, j] \leftarrow$ polynomial coefficient of f_j in $V[j]$.
  Set $Res_k \leftarrow VM$.
Return Res.

Time complexity: The time complexity of Procedure FindFencePPD is $\mathcal{O}(d^2 t)$, since we fill in the matrix M d times, where in each time it takes $d \cdot t$ time to fill it. Hence the total time complexity is $\mathcal{O}(d \cdot d \cdot t) = \mathcal{O}(d^2 t)$.

5. Imperfect detection

In many cases, even if the adversary passes through the sensorial range of the robot, it still does not necessarily detect it. Assume that the robot is currently monitoring some segment s_i. If the adversary penetrates through s_i while it is monitored, then the probability that the robot actually detects the adversary is p_d. The major difference when considering this case compared to the case in which $p_d = 1$, is that revisiting a segment can increase the probability of detecting the adversary. Therefore the probability of detection in a segment s_i (ppd_i) is *not* equivalent to the probability of *first* arriving at s_i, but the probability of

detecting the adversary during some visit m to s_i, assuming it was not previously detected. Denote the probability of the y'th visit of some robot to segment s_i by w_i^y. Therefore ppd_i is defined as follows.

$$\mathsf{ppd}_i = w_i^1 p_d + w_i^1(1-p_d) \times \{w_i^2 p_d + w_i^2(1-p_d) \times \{\ldots\{w_i^t \times p_d\}\}\} \quad (1)$$

In other words, the probability of detecting the penetration is the probability that it was detected in the first visit ($w_i^1 \cdot p_d$) plus the probability that it was *not* detected then, but in later stages. This again is the probability that it was detected in the second visit ($w_i^2 \cdot p_d$) or in later stages, and so on.

Note that in t time units, $w_i^t = 0$ for all segments s_i currently unoccupied by some robot, and if a robot resides in s_i, then this value is exactly $p_d(1-p)^t$.

Algorithm 2 Procedure $\mathsf{ComputeProbPPD}(d,t)$

1: Run Procedure $\mathsf{FindFencePPD}(d,t)$ while returning the entire polynomial, i.e., skipping line 16 of the procedure.
2: **for** $j \leftarrow 1$ to d **do**
3: **for** $i \leftarrow 1$ to d **do**
4: $w_i^y \leftarrow$ polynomial coefficient of f_i^y.
5: $PRes[j,i] \leftarrow$ substitution of $w_i^y, \forall y$ in Equation 1.

Theorem 1. *For each segment s_i, Procedure* FindFencePPD *computes* ppd_i.

Proof. In order to prove the theorem, we need to show that the algorithm computes correctly the probability of the m'th arrival to s_i, for every $1 \leq m \leq t$, i.e., the coefficient of f^m is exactly w_i^m. Since each path is multiplied by f each time it passes through s_i, then necessarily if the path went through s_i m times, it is a coefficient of f^m. By adding all arrivals to s_0^{cw} and s_0^{cc}, we take into consideration all visits to s_i, hence all paths going through s_i are taken into consideration, and by using Equation 1, we generate exactly ppd_i. □

As in FindFencePPD, the returned value is a matrix of size $d \times d$, where each row i contains d functions representing the ppd in each segment, given that the robot is currently in segment i.

6. Applying knowledge of ppd functions in determining the patrol algorithm

Finding the ppd in all segments of the section is the first step in determining the best patrol algorithm for the robots. After obtaining this information, a suitable function can be executed in order to meet the desired criteria which depends on the adversarial model. The criteria could be maximizing the minimal ppd for a strong adversarial model, or maximizing the expected ppd for a weak one [2].

In the case of a fence, the ppd value depends on the current location of the robot, hence the optimal p value characterizing the patrol of the robots is different for each segment s_i, where $1 \leq i \leq d$. Note that there could be different optimal p values with respect to both location and *orientation* of the robot ($2d$ values). However, it is enough to calculate the ppd values only d times - only for one direction, as the other direction is a reflecting image of the first (see Figure 3).

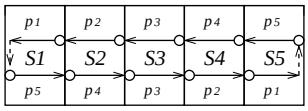

Figure 3. An illustration of the required output as a patrol algorithm, and where to use it.

6.1. Strong adversarial model - full knowledge

We demonstrate the use of Procedure FindFencePPD in the full knowledge adversarial model. In this model, when the adversary has to decide where to penetrate, it has full knowledge of the patrol algorithm and the current location of the roots. Therefore it will necessarily choose to penetrate through the segment in which the probability of penetration detection is minimal (the weak spot). Therefore in this case the robots should choose an algorithm that maximizes the minimal ppd along the section. For that we use algorithm FindP from [1] that finds the value p such that the minimal ppd is maximized. FindP computes this point by finding the maximal point in the integral intersection of all curves (ppd_i). The complete description of the algorithm is shown in Algorithm 3.

Algorithm 3 Procedure ComputeProbPPD(d, t)

1: $M \leftarrow$ FindFencePPD(d, t)
2: **for** $i \leftarrow 1$ to d **do**
3: $\quad OpP[i] \leftarrow$ FindP(d, t) [1] with given additional input $M[i]$ as vector of ppd functions.
4: Return OpP

6.2. Weak adversarial model - random penetration

We now turn to examine the case in which the adversary does not have full knowledge of the patrol algorithm. In this case, we assume the adversary chooses at random with uniform distribution its penetration spot from all unoccupied segments. Therefore we wish to maximize the *expected* ppd throughout the line.

In circular environments, it was proven that under this adversarial model, the optimal patrol algorithm is the deterministic algorithm. However, if the robot travels along a line - this is not the case any more. We show it by a counter example. The logic behind this fact is that in some segments, it is worthwhile to switch the direction of the robot before reaching the end of the section, and by that cover more segments.

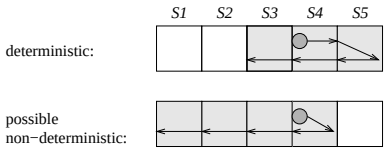

Figure 4. An illustration of a case in which the maximal expected ppd is obtained by a non deterministic algorithm. Each arrow represent a movement in one time cycle.

Consider the case in which $d = 5$ and $t = 5$. If the robot is placed in segment s_4, then the ppd values are as follows. $\mathsf{ppd}_1 = (1-p)p^4$, $\mathsf{ppd}_2 = (1-p)p^3$, $\mathsf{ppd}_3 = (1-p)p^2 + p^5 + (1-p)^3p^2$, $\mathsf{ppd}_4 = 1$ and $\mathsf{ppd}_5 = p + (1-p)^2p$, and the

expected ppd is maximized for $p = 0.418$. The logic, as explained previously, is illustrated in Figure 4. It demonstrates that the robot can profit from turning around at s_4, since it can reach four segments, where if it travels deterministically through all segments, i.e., go straight to s_5, it reaches only three segments.

This strengthens our need to find the actual ppd values in all segments, since after obtaining these functions, it is possible to calculate the p value that maximizes the expected ppd throughout the line. This can be done using a procedure similar to ComputeProbPPD, while replacing the call to FindP to a function that calculates the p value that maximizes the expected ppd.

7. Conclusions and future work

This paper discusses the problem of multi-robot patrol along an open polyline in adversarial settings. We show that this case is more challenging than the problem of multi-robot perimeter patrol, which is symmetric by its nature. We provide a polynomial-time algorithm for finding the probability of penetration detection in each segment along the line. We then tackle another challenge posed by previous studies, and show that this algorithm can be altered into considering also more realistic cases in which the robots do not have perfect detection, i.e., the robot will detect the adversary if it is in its sensorial range with probability p_d. We then demonstrate how the probability of penetration detection functions can be used in order to find the optimal patrol algorithm in various adversarial models.

There are various points we wish to address as future work. First, we are interested in incorporating more realistic considerations in the model, such that it will better suit various practical systems, for example sensing ahead of the segment with changing probabilities. Moreover, we intend to check the implications of the overlapping of sections on the optimal patrol algorithm in various adversarial models. We are interested in using the ppd functions in order to find game-theoretic solutions to the the problem. Last, we would like to see how this algorithm can be adapted to patrol in other domains (area patrol, for example).

References

[1] N. Agmon, S. Kraus, and G. A. Kaminka. Multi-robot perimeter patrol in adversarial settings. In *ICRA*, 2008.

[2] N. Agmon, V. Sadov, S. Kraus, and G. A. Kaminka. The impact of adversarial knowledge on adversarial planning in perimeter patrol. In *AAMAS*, 2008.

[3] M. Ahmadi and P. Stone. A multi-robot system for continuous area sweeping tasks. In *ICRA*, 2006.

[4] A. Almeida, G. Ramalho, H. Santana, P. Tedesco, T. Menezes, V. Corruble, and Y. Chevaleyr. Recent advances on multi-agent patrolling. *Lecture Notes in Computer Science*, 3171:474–483, 2004.

[5] H. Choset. Coverage for robotics—a survey of recent results. *Annals of Mathematics and Artificial Intelligence*, 31:113–126, 2001.

[6] J. Colegrave and A. Branch. A case study of autonomous household vacuum cleaner. In *AIAA/NASA CIRFFSS*, 1994.

[7] D. Coppersmith, P. Doyle, P. Raghavan, and M. Snir. Random walks on weighted graphs and applications to on-line algorithms. *J. ACM*, 40(3), 1993.

[8] Y. Elmaliach, N. Agmon, and G. A. Kaminka. Multi-robot area patrol under frequency constraints. In *ICRA*, 2007.

[9] Y. Elmaliach, A. Shiloni, and G. A. Kaminka. A realistic model of frequency-based multi-robot fence patrolling. In *AAMAS*, 2008.

[10] T. Haynes and S. Sen. Evolving behavioral strategies in predators and prey. In *IJCAI-95 Workshop on Adaptation and Learning in Multiagent Systems*, pages 32–37, 1995.

[11] P. Paruchuri, J. P. Pearce, M. Tambe, F. Ordonez, and S. Kraus. An efficient heuristic approach for security against multiple adversaries. In *AAMAS*, 2007.

[12] P. Paruchuri, M. Tambe, F. Ordonez, and S. Kraus. Security in multiagent systems by policy randomization. In *AAMAS*, 2007.

[13] R. Vidal, O. Shakernia, H. J. Kim, D. H. Shim, and S. Sastry. Probabilistic pursuit-evasion games: theory, implementation, and experimental evaluation. *Robotics and Automation, IEEE Transactions on*, 18(5):662–669, 2002.