Proactive Patrol Dispatch Surveillance System by Inferring Mobile Trajectories of Multiple Intruders Using Binary Proximity Sensors

Dahee Jeong[†], Minkyoung Cho[‡], Omprakash Gnawali[§], and HyungJune Lee[†]
[†]Ewha Womans University, South Korea
[‡]Korea Advanced Institute of Science and Technology, South Korea
[§]University of Houston, Texas, USA
Email: hyungjune.lee@ewha.ac.kr

***Abstract*—In this paper, we consider the problem of distributing patrol officers inside a building to maximize the probability of catching multiple intruders while minimizing the distance the patrol officers travel to reach the locations of the intruders. In our problem setting, the patrol officers are assisted by the information collected by a network of binary proximity sensors installed in the building. We claim that learning even common movement sub-patterns that originate due to the constrained physical environment helps to find likely locations of intruders where each major location is instrumented using a sensor node. We use a series of binary detection events to infer likely future trajectories in a real-world building. For a given set of detectable nodes on the inferred future trajectories, we aim to find the optimal patrol dispatch node location with high exposure to intruders' future appearance using patrol officers in limited numbers, ideally fewer than the intruders. In order to prevent possible crime and perform responsive defense against potential intruders, our algorithm also tries to reduce the travel distance from patrols current positions to their dispatched positions at the same time. We validate our proposed scheme in terms of detection accuracy by varying the number of intruders, robustness against missing events, and responsiveness compared to a practical baseline counterpart through real-world system experiments.**

I. Introduction

Building automation is a promising area of work that focuses on monitoring and controlling lighting and HVAC (Heating, Ventilating, and Air Conditioning) systems to drastically save total energy consumption by up to 60 – 70 % in typical buildings. Beyond the energy saving effort, devising an intelligent surveillance system with minimal human intervention is a crucial discipline to further explore in the era of Internet-of-Things (IoT).

In office buildings, there are usually only a limited number of security patrols. These patrols perform security checks in a regular manner at designated times or on a random intervals over somewhat regular paths. A smart intruder may take one of several minimum exposure paths [1] where one is unlikely to get caught. To protect building assets and casualties in buildings from intruders, a camera vision-based surveillance system [11] has been proposed. However, the camera vision-based surveillance faces the fundamental limitation of reactiveness. If patrols are dispatched to the place where the intruder was detected in the video, the intruder may already be somewhere else by the time the patrol reaches that place.

To address this problem in intruder detection and coverage in a more cost-effective manner, binary proximity sensor-based approaches have widely been proposed in [2], [9], [16], [23], [24], [27]. The prior works find monitoring areas with high exposure of intruders and obtain the movement scheduling strategies of patrols to efficiently traverse region of interests for detecting intruders. However, most of these prior work are theoretical and suffer from a practical limitation in directly applying to a real-world embedded system due to their strong assumptions of detection and mobility models.

Intruders' movement behavior usually includes deviant walking patterns (e.g., wandering-around, standing while being hidden), and it is believed that intrusion detection and prediction is a challenging and even infeasible problem. However, a previous study [25] provides a strong evidence that intrusion learning is feasible for passive intrusion detection. Its pattern profiling helps to increase intrusion detection probability. Our work also takes a similar methodology of intrusion learning.

In this paper, we present a patrol dispatch surveillance system based on a network of binary proximity sensors in a real-world testbed. We exploit the inferred knowledge of future movement of intruders to dispatch patrols to security spots that have high exposure to as many intruders as possible. We aim to maximize the detection probability while minimizing the travel distance of patrols to intruders.

We first construct a database of all possible classic trajectory patterns covering travel paths in the network based on binary detection events from normal non-intruders in the off-line learning phase. In the execution phase, if an intruder is detected over a series of binary proximity sensors along his somewhat deviating travel path, our system puts together several best pattern matching clusters that include partial but similar travel patterns in their subsets through dynamic programming. Based on this information, we obtain all likely detectable nodes which the intruder would travel in the near future with high probability.

Once we infer the likely detectable nodes with their detectable probabilities, we aim to find the best reachable node

regions through a stochastic optimization. We formulate the problem of patrol dispatch into a binary integer program that attempts to maximize the coverage of patrols to find intruders, while reducing travel distance.

We evaluate our proposed surveillance system in terms of detection accuracy and responsiveness in a real-world testbed compared to a baseline algorithm. Our work can effectively detect intruders with few patrols, and the detection accuracy does not decrease with increasing number of intruders. More importantly, even under severe detection failures on binary proximity sensors, our system demonstrates its robustness, satisfying an important practical aspect.

It should be noted that our work is not the one that claims to predict the future deviant patterns of intruders one-on-one, but the one that locates some highly detectable areas that appear more commonly over the inferred future trajectories.

To the best of our knowledge, this paper is the first to implement a practical patrol dispatch algorithm based on an applicable stochastic optimization in a real-world testbed.

The rest of this paper is organized as follows: After discussing related work in Sec. II, we provide an overview of our proposed system in Sec. III. In Sec. IV, we present our inference algorithm for estimating future detectable nodes of intruders, and Sec. V proposes our dispatch optimization algorithm. After demonstrating real-world validation of our approach in Sec. VI, we conclude this paper in Sec. VII.

II. Related Work

The intruder detection and coverage problem have been explored in the wireless sensor network research community. Most of prior works focus on finding monitoring areas with high exposure of intruders [2], [9], [23], or scheduling the patrols to efficiently traverse region of interests for detecting intruders [13], [16], [24], [26], [27].

Previous works on extracting region of interests for finding intruders have been studied in sensor networks and surveillance camera networks. In [2], a centralized online learning algorithm is proposed to monitor transmission activities in multi-channel wireless networks by performing the optimal channel assignment for wireless surveillance. Wu et al. [23] propose a real-time video surveillance system with an automated camera control toward region of interests by keeping track of moving targets. The authors in [9] try to extract the most important event areas among all of the event areas to monitor subject to resource constraints. Given all possible event areas, they formulate the device activation problem to select few event areas out of them for wakeup into an online integer linear program solvable within a polynomial time.

There have also been research in the security patrolling problem in which security agencies should deploy patrols and checkpoints to protect from terrorists [13] and illegal free riders in public transportation [26] currently applied for fare inspection in the Los Angeles Metro Rail system. These advanced patrolling strategies are devised based on a game-theoretic perspective. Both [13] and [26] formulate the problem of patrolling scheduling into a Stackelberg game between a leader (the defender) and a follower (the adversary). A successful patrol's strategy is known to mix a randomized schedule over deterministic schedules.

In robotics, similar issue has been explored as *search and pursuit-evasion* [6]. The problem is formulated into pursuit-evasion games where the pursuers aim to catch the evaders, while the evaders try to avoid the detection from the pursuers. Prior works [12], [18] devise a conservative strategy of pursuers that maximizes search performance in the worst case scenario from theoretical perspectives. For example, it is assumed that the evaders have full knowledge of pursuers' behaviors and environments (e.g., plane, grid, graph, and polygons). As another stream of the pursuit-evasion research, incorporating probabilistic approaches [14], [20] provides the average-case performance of the pursuit strategy.

More closely to our work, previous literature [16], [24], [27] explores dynamic deployment of mobile sensors (or patrols) to minimize the total travel distance while achieving high detection probability. [27] presents a dynamically evolving sensor deployment strategy with a probabilistic scoring-based localization algorithm to maximize the sensor field. The work assumes only mobile sensors in the network, and a simple binary disk model or a still theoretical probabilistic model for the target detection. Prior works [16], [24] extend the problem with the usage of both static and mobile sensors for intruder detection. [24] presents a multi-sensor fusion model consisting of two phases where in the first phase, all of sensors periodically send the measurements from environment to a cluster head, and in the second phase, the cluster head decides a set of mobile nodes to move toward the surveillance position. This work assumes that intruder targets send a signal, and nearby sensors can measure the energy of the signal transmitted by the targets for detection. The proposed model is validated with somewhat unrealistic environments such as random moving speed of mobile nodes and the theoretical signal-based target sensing model. Similarly, in [16], under the sensor location model of stationary two-dimensional Poisson point process and a variant of the random waypoint mobility model, the authors formulate the dynamic coverage problem and solve a mixed strategy Nash equilibrium.

However, most of the above related works take theoretical approaches by constructing analytical models and proving the feasibility of the proposed algorithms in proofs or simulations. Therefore, their works have some limitations in practice due to their strong assumptions in the intruder detection model and the mobility model of mobile sensors. There has been a practical mobile surveillance and wireless sensor embedded system called iMouse [21] that dispatches mobile sensors to detected regions by static sensors. We use this algorithm as a baseline counterpart to our work in evaluation. Our proposed work goes beyond theoretical prior works: Based on real-world mobile traces with real-world detection traces using binary proximity sensors, our work leverages the inferred knowledge of future movements of intruders to determine where to dispatch patrols to increase detection probability while reducing travel distance.

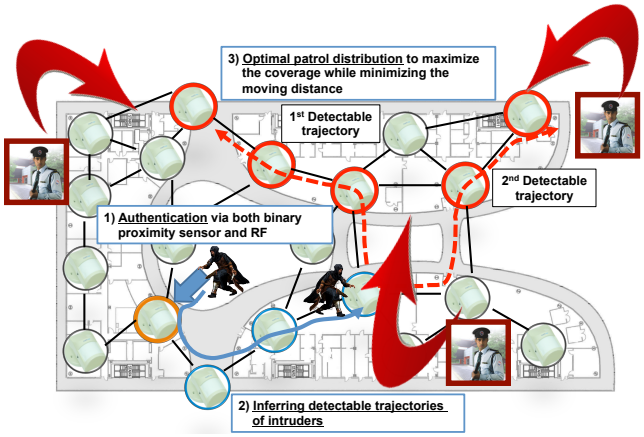


Fig. 1. Overall procedure of our proposed surveillance system.

III. Overview

This work designs and evalutes algorithms to autonomously distribute patrols in a building to catch multiple intruders. Furthermore, we want the algorithm to work in buildings covered by a limited number of patrols using a network of binary proximity sensors. We consider a static one-time dispatch decision problem where patrols (humans or robots) are dispatched to the selected security checkpoints on standby until some of intruders come around. The main objective of this paper is to maximize the detection probability while minimizing the travel distance of patrols to the intruders. We use a series of binary detection events to infer possible future trajectories in a building. For a given set of detectable nodes on the inferred future trajectories, we aim to find the optimal patrol dispatch positions to maximize the coverage of patrols toward intruders particularly when the number of patrols is less than the number of intruders. In order to prevent possible crime and perform some responsive defense against potential intruders, we try to reduce the travel distance from patrols' current positions to their dispatched positions.

Our system consists of binary proximity sensors installed in the hallways of a building. An example of such a sensor is a passive infrared (PIR) sensor that can detect the presence of moving objects. The sensors form an ad-hoc network covering all the possible moving paths in a building. Each sensor node can deliver detection events to a collection server over multi-hop routing. Also, it is assumed that we can differentiate multiple intruders from one another, relying on seminal multiple target tracking algorithms [7], and thus, the problem of multiple target tracking is out of scope in this paper. Instead, we focus more on how the inferred knowledge for intruders can contribute to improving the patrol distribution strategy in terms of detection probability and travel distance.

A. Procedure

We consider a scenario such that intruders enter a building and roam around a building in Fig. 1. Our proposed surveillance system starts operating as soon as a potential intruder is detected somewhere in the building. The proposed system

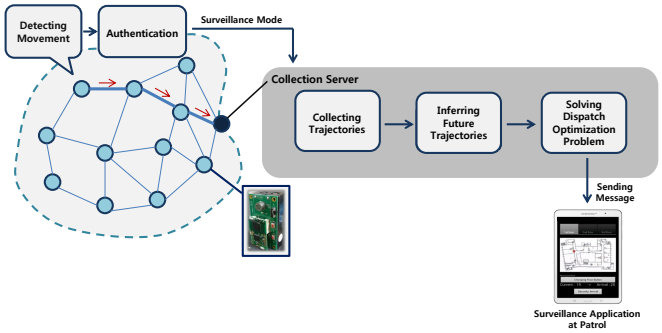


Fig. 2. System overview.

in Fig. 2 performs a proactive automated decision of the patrol distribution, based on the following three phases as in Fig. 1.

1) Authentication: Our surveillance system starts with an authentication procedure to check whether a detected person is a valid registered user in the building. Our system is agnostic to the exact mechanism used for authentication. The only requirement is legitimate users have a way of informing the surveillance system that they are indeed legitimate users. Possible mechanisms could include swiping the card at the point of entry, or continuous authentication schemes such as face scan using surveillance camera. The mechanism could also include legitimate user carrying a special radio (as part of their badge) that authenticates with the building systems as they work and walk in the building. If the system detects a moving person without successful authentication, the system triggers surveillance mode in the network by sending a message to the dispatch center using a collection protocol, where we run the dispatch algorithm by gathering sensor data from the building.

2) Inferring the future detectable trajectories of intruders: When the surveillance mode is enabled, all of the deployed sensor nodes report any movement detection event with their own node IDs to a central server using a collection protocol, e.g., CTP (Collection Tree Protocol) [10] in TinyOS 2.x. The central server continuously collects the reported events and records a series of node IDs into a chronological sequence. As the sequence is accumulated with up to a certain length, we attempt to search over the mobile trajectory database that has been constructed in the learning phase, and find several best matching pattern clusters. Based on a set of the probable matching patterns, we infer the possible detectable trajectories in terms of node ID where intruders will likely pass through and be detected accordingly in the near future. We describe a detailed procedure and algorithms of constructing mobile trajectories using binary proximity sensors and inferring the future detectable trajectories of intruders in Sec. IV.

3) Dispatching patrols: Once a set of all possible detectable trajectories of multiple intruders are inferred, we find the optimal dispatch positions of available patrols in the network. We aim to maximize the coverage of patrols toward intruders (especially where the number of patrols is less than the number of intruders), while trying to minimize the travel distance to the dispatched position to achieve patrols' responsiveness. By considering possible detectable nodes with their relative detectable importance (in probability) to cover more intruders

and travel distance to each corresponding node, we solve a patrol dispatch optimization problem. We present how to formulate the problem with constraints into a binary integer program and solve it so that the system can notify patrols of dispatch locations that maximize the exposure toward intruders in the network in Sec. V.

IV. Inferring Mobile Trajectories

In this section, we present lightweight yet efficient algorithms of constructing mobile trajectory database using only binary detection events from binary proximity sensors, and eventually inferring the future detectable trajectories. Sufficient amount of binary detection data even without the location information of deployed sensors can extract reasonably accurate mobile trajectories in indoor buildings [5], [22]. Beyond the underlying observation, we aim to infer all possible detectable nodes where intruders will pass by soon with high probability, using a sequence of past detected node IDs with a certain length.

Our proposed scheme consists of two phases: (1) off-line learning phase and (2) on-line execution phase. During the off-line learning phase, we construct a mobile trajectory database consisting of classified trajectory pattern clusters that cover all the possible moving paths in a building. It should be noted that during the learning phase, we aim to extract a set of *common* trajectory segments from normal non-intruders. In the execution phase, when a potential intruder has been detected by some number of binary proximity sensors, the central server will use these information collected by triggered nodes to infer probable detectable nodes by searching over the database and finding matching clusters. As intruders tend to deviate from common movement patterns (while still constrained by physical environment with some limited degree of freedom), we use several pattern clusters together to predict security checkpoints with high detectability.

We use only the sensor node IDs that detect moving objects without using timestamp to obtain trajectory sequences in the learning phase, and also a trajectory sequence of each intruder. The reason is that intruders' movement may include somewhat roam-around patterns with indeterministic stay time at certain areas, and thus learning visiting time patterns from common movements in the learning phase would rather undermine the prediction for intruders' deviating movements in the execution phase.

A. Mobile Trajectory Database

We present an efficient procedure of constructing a database of all possible trajectory patterns based on binary detection events from the network. In the off-line learning phase, a mobile user explores possible moving trajectories, and binary proximity sensors that detect the user's existence send reports to a central server through a data collection protocol. For a given physical moving path of the user, we obtain a corresponding trajectory sequence that chronologically records only node IDs that have detected the user. By exploring all possible moving paths, the central server can collect all corresponding unlabeled trajectory sequences.

Once the unlabeled trajectory sequences are collected, we perform a clustering procedure to make several representative trajectory groups where similar sequences can belong to a group. To build a mobile trajectory database, the procedure is divided into two steps.

1) Calculating pairwise distance between trajectories: To calculate the pairwise distance between trajectory sequences, we apply the edit distance to provide different penalty for insertions, deletions, and substitutions between two sequences.

$$D(i,j) = \begin{cases} D(i-1,j-1) & \text{if } X_i = Y_j \\ \min \begin{cases} D(i-1,j) + w_{ins} \\ D(i,j-1) + w_{del} \\ D(i-1,j-1) + w_{sub}(X_i, Y_j) \end{cases} & \text{otherwise} \end{cases} \quad (1)$$

To transform a sequence of $X[1,\ldots,m]$ into another sequence of $Y[1,\ldots,n]$, the insertion cost of w_{ins}, the deletion cost of w_{del}, and the substitution cost of $w_{sub}(X_i, Y_j)$ are incurred. We calculate the edit distance between X and Y as $D(m,n)$.

However, since the original measure of edit distance tends to be distorted as the length of one of sequences increases in our mobile trajectory application, we modify it into a distance measure with normalization over the length of the shorter sequence as follows:

$$\overline{D}(m,n) = \frac{D(m,n)}{\min(m,n)} \quad (2)$$

To make the commutative property hold for the distance measure, we use $w_{ins} = w_{del}$. Our real-world experiments prove that the parameters of $w_{ins} = w_{del} = 1$ are effective, achieving the highest intra-cluster similarity, and $w_{sub}(X_i, Y_j)$ should be given with respect to network topology where the chosen value ranges in [1, 8]. It should be noted that as X_i is located further away from Y_j in network topology, the value of $w_{sub}(X_i, Y_j)$ increases.

2) Clustering and producing cluster signature: After calculating all the pairwise distances of trajectory sequences collected during the learning phase, we categorize the sequences into a fixed number of trajectory clusters. We apply a classic hierarchical clustering algorithm to build the trajectory hierarchy from the individual trajectory sequences [17]. To find the optimal number of clusters, we probe the slope of the average and the standard deviation of intra-cluster sequence distance and choose the number of clusters where the slope is abruptly changed and then becomes relatively flat afterward. It means that increasing the number of clusters beyond this point does not significantly contribute to reducing the average pairwise distance inside each cluster any longer.

When the optimal number of trajectory clusters is found, and similar trajectories are grouped into a fixed number of clusters, we produce a *cluster signature* S per trajectory cluster. It facilitates efficient search for finding matching clusters for a given test sequence in the execution phase. Instead of exhaustibly finding relevant mobile trajectories against all of

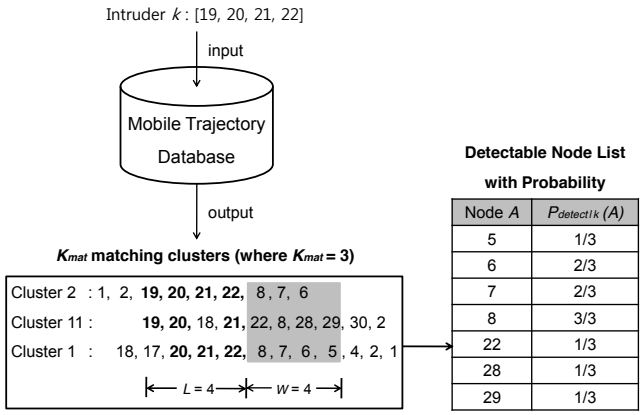


Fig. 3. Inference procedure of future detectable nodes from matching clusters for intruder k.

the training sequences for a given test sequence, the only thing we need to perform is to compare it to only K cluster signatures (where the number of clusters is turned out to be K), considerably reducing computational complexity.

To create a simple yet efficient cluster signature for given clustered sequences per cluster, we use the longest common subsequence of clustered sequences as the cluster signature where the total computational complexity is given by $K \cdot O(n)$ where n is the number of trajectory sequences in a cluster. This is one of the key advantages of our approach in terms of algorithm running time as opposed to related previous works on making cluster representations usually accompanied with multiple sequence alignment techniques taking intensive computations [8], [15]. Lowering computation complexity to build compact representations for clusters is even more important in case that the database needs frequent updates for more dynamic scenarios.

B. Inferring Future Trajectories

In the execution phase, if a potential intruder has been being detected by binary proximity sensors in the network, and the resulting sequence with the length of L is collected at the central server, our proposed method starts inferring possible future detectable trajectories by finding matching clusters for the given sequence $X'[1, \ldots, L]$. We present a new cluster matching algorithm that calculates a matching score between a cluster signature S and an online sequence X' based on dynamic programming as follows:

$$M(i,j) = \begin{cases} M(i-1, j-1) + w_{mat} & \text{if } S_i = X'_j \\ \max \begin{cases} M(i-1,j) & \text{otherwise} \\ M(i,j-1) & \\ M(i-1,j-1) + \alpha / w_{sub}(S_i, X'_j) & \end{cases} \end{cases} \quad (3)$$

where w_{mat} is the matching score for the exact matching case.

In this dynamic programming, we incur the highest score for the exact matching (if $S_i = X'_j$) and the second highest score for the partial matching by also taking into account the distance measure between S_i and X'_j, i.e., $w_{sub}(S_i, X'_j)$. It implies that the more similar a cluster signature and a sequence are, the higher the matching score is. Based on this measure, we choose the best matching K_{mat} clusters (out of the total K clusters) to infer them as relevant trajectories with similar patterns for an intruder. The reason that we use not only the best matching cluster, but a set of several matching clusters as well is to cover somewhat uncommon moving patterns deviating from certain common moving patterns based on the observation that intruders tend to wander around from place to place.

After selecting K_{mat} matching clusters, we divide each matching signature into the matching part, and the future detectable part coming after the matching part. For a given matching cluster's signature $S[1, \ldots, m]$ and the online sequence $X'[1, \ldots, L]$, we find the ending index k of the matching part that satisfies the following condition:

$$M(k-1, L) < M(k, L) = M(k+1, L) = \ldots = M(m, L).$$

Accordingly, we obtain the detectable node list $S[k+1 : m]$ for a matching cluster S. In this way, we collect all the detectable nodes over the K_{mat} clusters with the detectable probability of how often a node appear over the future W nodes of K_{mat} clusters' signatures, i.e., $P_{detect|k}(A)$ at node A for intruder k as illustrated in Fig. 3 where W is the number of future trajectory nodes. Finally, we use them as all the inferred future trajectory nodes together with their detectable probabilities for a given potential intruder.

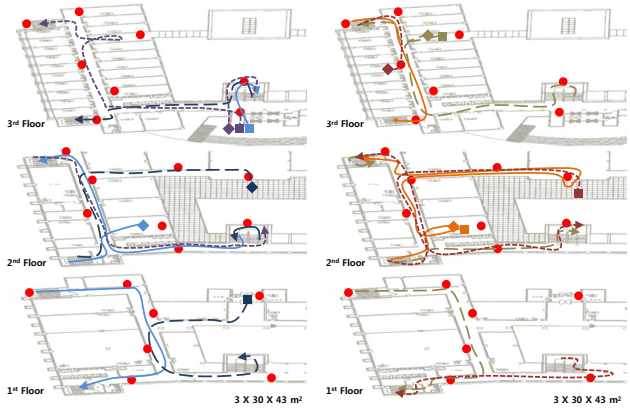
V. Optimal In-Network Dispatch Algorithm

Given the likely detectable nodes with their detectable probabilities, we exploit the inferred knowledge of intruders' future movements to investigate the best reachable positions in a stochastic optimization sense. We present an optimal in-network dispatch algorithm of finding dispatch positions of patrols in the network that maximizes the coverage of patrols to intruders, while achieving responsiveness by reducing travel distance. One of the advantages of our proposed algorithm is that it considerably benefits security surveillance efficiency especially where the number of patrols is less than that of intruders.

We formulate the problem of patrol dispatch into a binary integer program. Our goal is to find the optimal dispatch node to which each patrol needs to be dispatched respectively for covering as many intruders as possible for a given a limited number of patrols. Concurrently, we want to minimize the travel distance from a patrol's initial position to the selected dispatch node to reduce travel time of patrols.

We introduce indication functions $I^{(k)}_{i \to A_j}$ denoting whether patrol i should be dispatched to node A_j for detecting intruder k where $1 \le i \le N_P$(= the number of patrols), $1 \le j \le N_B$(= the number of binary proximity sensors), and $1 \le k \le N_I$(= the number of intruders). Additionally, we introduce other indicator functions $J_{i \to A_j}$ indicating whether patrol i is dispatched to node A_j.

Based on this notations, we define the objective function to maximize the detectable probability over the travel distance to the selected node as



(a) 3 exemplary training routes out of total 31 different routes.

(b) 3 exemplary testing routes out of total 10 different routes.

Fig. 4. Real-world testbed network and moving routes for experiments where ■: the starting point of a route, ⧫: the ending point of a route, and ●: PIR sensor node.

$$\text{maximize} \quad \sum_{i,j,k} I^{(k)}_{i\to A_j} \cdot \frac{P_{detect|k}(A_j)}{f(dist(i \to A_j))} \tag{4}$$

$$\text{subject to} \quad \sum_{i,j} I^{(k)}_{i\to A_j} \geq 1 \qquad \forall k \tag{5}$$

$$I^{(1)}_{i\to A_j} = I^{(2)}_{i\to A_j} = \ldots = I^{(N_I)}_{i\to A_j} \quad \forall i, \quad \forall j \tag{6}$$

$$I^{(k)}_{i\to A_j} \leq J_{i\to A_j} \tag{7}$$

$$\sum_{j} J_{i\to A_j} = 1 \qquad \forall i \tag{8}$$

where $P_{detect|k}(A_j)$ is the probability of detecting intruder k at node A_j, and $f(dist(i \to A_j))$ can be any kind of distance measure from the node where the patrol i is currently located, to node A_j. It should be noted that the function f is a non-decreasing function of $dist(i \to A_j)$. By penalyzing a long travel distance from patrol i's current position to the selected dispatched node A_j in the objective function, this formulation implicitly takes into account travel time information.

Constraint (5) ensures that the patrol dispatch should be considered for every single intruder. Constraints (6) – (7) force the dispatch event of patrol i to node A_j to potentially detect every intruder. The last constraint (8) enforces each patrol to be dispatched to one of nodes in the network.

Finally, we obtain a solution of *which security patrol* should be dispatched to *which sensor node* for maximizing the objective function Eq. (4), while satisfying constraints Eq. (5) – (8). The above binary integer program is solvable based on the branch-and-bound methods [3]. We use a practical programming solver, `bintprog` Optimization Toolbox in MATLAB to obtain the optimal solution.

VI. Evaluation

We validate our proposed patrol dispatch algorithm by implementing a whole embedded system consisting of binary proximity sensors, a collection server, and Android devices. We construct a real-world testbed in a university campus building. We use TinyOS 2.1 and deployed 23 passive infrared

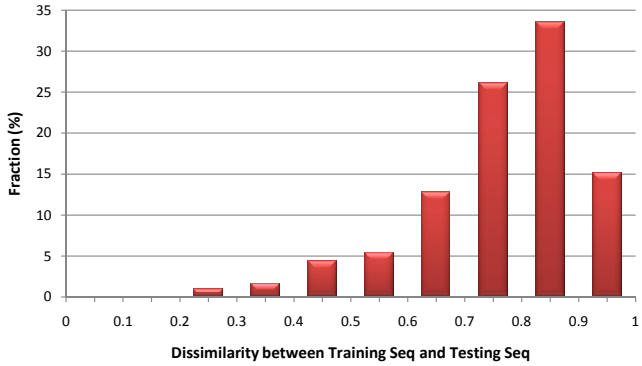


Fig. 5. Dissimilarity distribution between a training sequence and a testing sequence in datasets.

sensors (attached to TelosB-variant motes) over three floors in a $3\times30\times43$ m^2 area at a real-world office building as in Fig. 4. We placed them by keeping the regular distance among sensor motes less than 26 m, resulting in the high link quality of over 90 % and having no isolated sub-network under 0 dBm transmission power.

We implemented the proposed algorithms of inferring mobile trajectories and optimally dispatching patrols on an embedded hardware platform. The stationary binary proximity sensors were programmed to detect any nearby movement and deliver the detection event with the sensor node ID to a collection server through a sensor mote attached to it. We use the state-of-the-art data collection protocol, CTP [10] as the underlying collection algorithm. The algorithm of inferring mobile trajectories runs on the collection server on a MSI FX61 PC with Ubuntu Linux 12.04 LTS, 2.4 GHz Core 1 Quad processor, and 1 GB RAM. The server invokes the bintprog utility in MATLAB to solve our dispatch algorithm. Once all of dispatch nodes are determined for each patrol, the resulting decision is distributed to the respective patrol via a notification message at our Android application.

We obtained a training set of 31 different routes, each of which we recorded 3 times as in Fig. 4(a) and a testing set of 10 different routes as in Fig. 4(b). None of the training routes and the testing routes are from the same physical path, and only few partial path segments overlap among them. In addition, to simulate intruders' uncommon movement behaviors, the testing sequences are relatively longer than the training sequences with a factor of 1.9 on average, while also including wandering-around movements in the subsets of testing sequences. To quantify how well our testing sequences reflect intruders' uncommon movement pattern, we measure how testing sequences (for intruders) are deviant enough in movement compared to training sequences in terms of the longest common subsequence. To show dissimilarity between a training sequence and a testing sequence, we subtract the length of the longest common subsequence from the length of testing sequence with the normalization. Fig. 5 shows that each testing sequence includes uncommon subsequences with a substantial portion of 0.77 compared to each training sequence.

In our experiments, the function $f = \gamma \cdot dist(i \to A_j)$ is chosen where γ = 10 and $dist(i \to A_j)$ is the hop distance

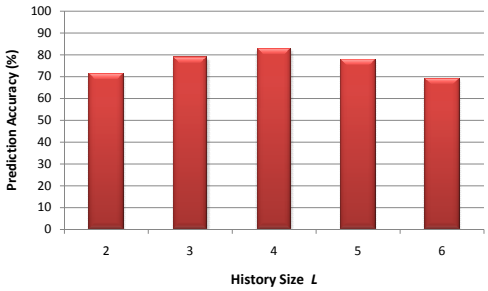


Fig. 6. Prediction accuracy with respect to the history size L used for inferring future mobile pattern.

from the currently located node of patrol i to node A_j. The parameters of L = 4, W = 4, K_{mat} = 3, $w_{ins} = w_{del} = w_{mat}$ = 1, α = 1 are tuned to use, unless otherwise noted.

We evaluate prediction accuracy of the inferred detectable nodes by varying parameters in Sec. VI-A, and then investigate how our dispatch algorithm using the predictive knowledge performs in terms of detection accuracy and travel distance in Sec. VI-B.

A. Detectable Node Prediction

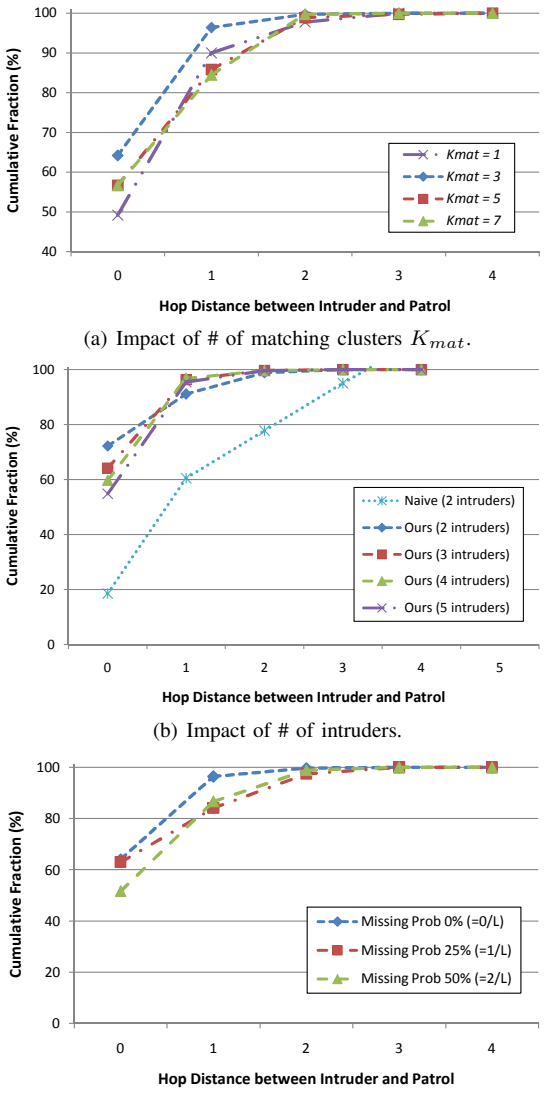
We investigate prediction accuracy of detectable nodes with respect to history size L (as defined in Sec. IV-B) used for inferring future trajectories. The prediction accuracy is defined as how well the predicted trajectory nodes from the trained database actually cover ground-truth nodes in each testing route. As Fig. 6 shows, using the history size of 4 provides the highest prediction accuracy reaching up to 83 %. This implies that a series of four consecutive detection events using binary proximity sensors are enough to characterize a specific mobile pattern in the network and accordingly provide a critical clue for inferring highly probable future mobile patterns. Also, this demonstrates an important claim that learning even common patterns of non-intruders can contribute to the correct prediction of future passing-by nodes of intruders with uncommon movements in parts.

B. Dispatch System Performance

Given the prediction accuracy of detectable nodes, we explore system performance of our dispatch algorithm that uses the predicted detectable nodes as input. The performance metrics are (1) how close dispatched patrols are located from intruders and (2) travel distance from the currently located node of a patrol to the dispatched node. To derive more general results that can be applied for an even more dense or sparse deployment of binary proximity sensors, we show the number of hop distance on the path instead of physical distance.

We compare our algorithm against a heuristic dispatch algorithm [21] that attempts to dispatch mobile sensors to static sensor nodes that have detected the event. To enable the algorithm to work with a larger number of intruders than a given number of patrols (or mobile nodes), we improve the algorithm by dispatching patrols to highly likely detectable nodes over the latest L detections of intruders for fair comparison with our algorithm. We call it as *Naive* algorithm for a baseline counterpart.

We evaluate dispatch accuracy of how accurately the dispatched patrols can detect intruders. We measure the shortest



(a) Impact of # of matching clusters K_{mat}.

(b) Impact of # of intruders.

(c) Impact of missing events for robustness (L=4).

Fig. 7. Dispatch accuracy in terms of hop distance between dispatched patrol and intruder.

traveling hop distance from the selected dispatch node to one of intruders' future W ground-truth nodes to show the coverage of patrols toward intruders in the network. In our experiments, we initially locate two patrols at certain nodes in the network and three intruders roaming over a subset of 10 different testing routes in the network, unless otherwise noted.

First, we explore the impact of selecting the number of matching clusters K_{mat} for prediction on dispatch accuracy in Fig. 7(a). When we use three matching clusters in the process of inferring future trajectories and apply the resulting detectable nodes for our dispatch algorithm, it shows the highest performance. This means that employing three different movement patterns to cover some deviating trajectory of intruders provides very important information to estimate where they will be in the near future. It should also be noted that the optimal K_{mat} value may be different depending on variety and irregularity of intruder's moving paths in different testbeds and needs to be optimized in the learning phase.

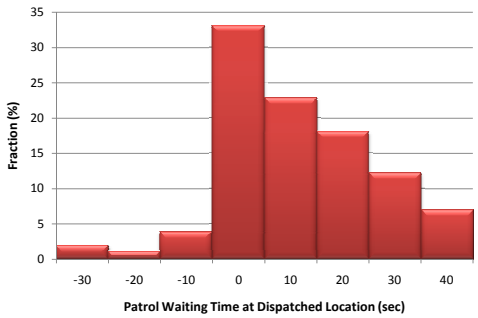


Fig. 8. Patrol waiting time at dispatched nodes until intruders come around.

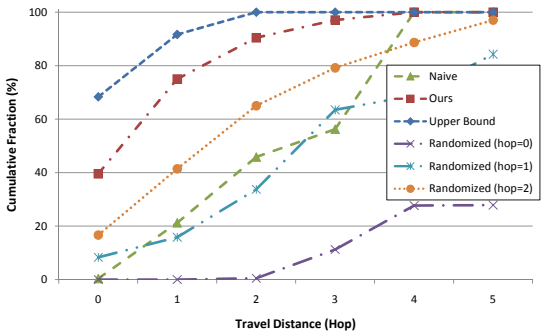
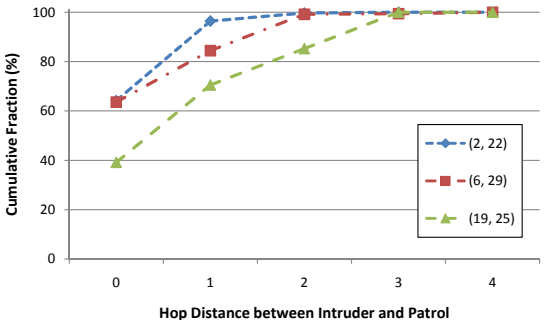
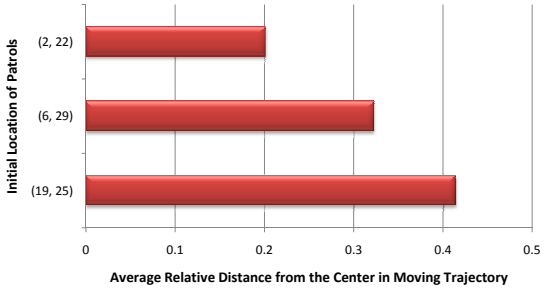


Fig. 9. Travel distance of patrols with comparison of naive, our algorithm, the upper bound, and a randomized patrolling.



(a) Dispatch accuracy with respect to initial location of patrols.



(b) Average relative distance from the center of moving trajectories on dispatching accuracy with respect to initial location of patrols.

Fig. 10. Impact of initial location of patrols on dispatching accuracy.

We investigate how the number of intruders N_I affects dispatch accuracy by varying it as in Fig. 7(b). For a given N_I, we choose all possible N_I testing routes out of the total 10 different testing routes for quantitative analyses. The performance of our algorithm gets only slightly worse when we increase the number of intruders, whereas *Naive* algorithm shows the worst performance even with only two intruders. This demonstrates that our algorithm fully leverages a limited number of patrols to cover an even larger number of intruders by finding out crucial security locations.

Also, we quantify the robustness of our surveillance system. In real-world surveillance systems, there should be missing detection cases due to PIR sensor's narrow detection angle and various hardware failures. We investigate how our algorithm survives even under frequent missing event scenarios. For this purpose, we try to intentionally omit each detected node with a certain probability from each testing sequence before applying our prediction algorithm. After collecting L detected nodes in the online execution phase, we run our proposed algorithm to measure dispatch accuracy. As Fig. 7(c) shows, even under the half-and-half missing detection events, our algorithm maintains relatively high performance and demonstrates the robustness of our surveillance system in a real-world environment. Interestingly, this result can shed a different light on proving a practical combination with sleep/wakeup mechanism very effective for energy saving.

Next, we measure how long the dispatched patrols have to wait for intruders to arrive at the patrol's dispatch location. To calculate the patrol waiting time, we perform the deduction of the number of traveling hops for patrols to dispatch to the security checkpoint nodes from the number of traveling hops for an intruder to one of them in our test dataset. Based on our measurement, the average walking time between two sensor motes is 10 seconds where the sensor motes are less than 26 m apart. Thus, the walking speed is about 2.6 m/s. We obtain the patrol waiting time with an approximation as in Fig. 8, showing that patrols can encounter intruders sooner than 30 seconds 73% of the time. In less than 7% cases, the intruders would have already passed by when the patrol reaches the dispatch location. Thus, in most cases, the patrol can catch the intruder based on predicted trajectory.

We also evaluate travel distance of patrols from their initially located nodes to the selected dispatched nodes to show the responsiveness of our system. To understand the travel distance achievable by our algorithm, we find a theoretical limit of travel distance. We choose the closest node to which each patrol is dispatched from its initial location where intruders will visit soon assuming the perfect future knowledge, irrespective of their resulting detection probability. This theoretical limit is optimized only for travel distance, called with *Upper Bound*. We also implement a randomized patrol strategy based on widely used real-world security patrolling [13], [26], as another baseline. The *Randomized* strategy lets patrols move to the next node that is selected randomly among uncovered nodes, and finish their movements until patrols encounter all of intruders. As in Fig. 9, our algorithm locates patrols to dispatched location from their initial location within only two hops (corresponding to approximately less than 52 m) with the percentage of over 90 %. It should be noted that the performance of our algorithm is relatively close to that of *Upper Bound* algorithm, whereas *Naive* and *Randomized* algorithms take much longer travel distance for the patrols. Even if we make the randomized algorithm optimize its travel

distance by admitting the patrol-intruder encounters as long as they are few hops away each other, our prediction-based approach outperforms the randomized approach. This validates the responsiveness of our surveillance system.

Lastly, we study how the initial location of patrols affects dispatch accuracy in our surveillance system. Given the different initial location of two patrols in Fig. 10(a), it turns out that dispatch accuracy is severely affected. This implies that choosing patrols' starting point is very important to determine our surveillance system performance. To understand how the initial location can affect the fidelity, we calculate the relative distance in [0, 0.5] of how the initially located nodes are far from the center of moving trajectory in our mobile trajectory database constructed during the off-line learning phase. It turns out that as the initial location of patrols is near the center of moving trajectory on average in the mobile trajectory database, more effective dispatch node locations are likely selected for covering more intruders with high fidelity as in Fig. 10(b).

VII. Conclusion

We have presented a practical embedded surveillance system based on a network of binary proximity sensors. The proposed system consists of the following two main algorithms: (1) inferring possible future moving paths in terms of detectable nodes in trajectory based on dynamic programming and (2) running a stochastic optimization algorithm to compute the optimal dispatch location for maximizing detection probability over intruders, while reducing the travel distance for patrols.

Our experiments based on embedded system implementation in a real-world testbed indicate that our system is able to detect many intruders even with a less number of patrols, and is affected less as the number of intruders increases as opposed to baseline counterpart algorithms. More importantly, even under severe detectability degradation on binary proximity sensors, our system maintains high accuracy. If duty cycle MACs [4], [19] are to be combined with our algorithm, detection accuracy would not be sacrificed much, while instead obtaining the benefit of energy saving. We also demonstrate the responsiveness of our system with short travel distance of patrols. Our surveillance system fully leverages only a limited number of patrols to cover an even larger number of intruders by finding out crucial security locations.

For future work, we would devise a distributed algorithm that uses only local detection events to compute a reasonably accurate dispatch strategy, taking even less computation complexity and communication overhead for improved scalability. Also, by employing energy saving techniques such as duty-cycle MACs into a surveillance system, exploring a trade-off relationship between system fidelity and energy saving would be a promising research direction.

Acknowledgment

This work was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(NRF-2015R1D1A1A01057902), and by the Ministry of Science, ICT, and Future Planning(NRF-2013R1A1A1009854).

References

[1] E. Amaldi, A. Capone, M. Cesana, and I. Filippini. Coverage planning of wireless sensors for mobile target detection. In *IEEE MASS*, 2008.

[2] P. Arora, C. Szepesvári, and R. Zheng. Sequential learning for optimal monitoring of multi-channel wireless networks. In *INFOCOM*, 2011.

[3] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

[4] M. Buettner, G. V. Yee, E. Anderson, and R. Han. X-mac: a short preamble mac protocol for duty-cycled wireless sensor networks. In *ACM SenSys*, pages 307–320, 2006.

[5] E. Cho, K. Wong, O. Gnawali, M. Wicke, and L. Guibas. Inferring mobile trajectories using a network of binary proximity sensors. In *IEEE SECON*, pages 188–196, 2011.

[6] T. H. Chung, G. A. Hollinger, and V. Isler. Search and pursuit-evasion in mobile robotics. *Autonomous robots*, 31(4):299–316, 2011.

[7] O. Demigha, W.-K. Hidouci, and T. Ahmed. On energy efficiency in collaborative target tracking in wireless sensor network: a review. *Communications Surveys & Tutorials, IEEE*, 15(3):1210–1222, 2013.

[8] R. C. Edgar. Muscle: multiple sequence alignment with high accuracy and high throughput. *Nucleic acids research*, 32(5):1792–1797, 2004.

[9] X. Fang, D. Yang, and G. Xue. Strategizing surveillance for resource-constrained event monitoring. In *IEEE INFOCOM*, 2012.

[10] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis. Collection Tree Protocol. In *ACM SenSys*, November 2009.

[11] S. Hengstler, D. Prashanth, S. Fong, and H. Aghajan. Mesheye: a hybrid-resolution smart camera mote for applications in distributed intelligent surveillance. In *ACM/IEEE IPSN*, pages 360–369, 2007.

[12] V. Isler and N. Karnad. The role of information in the cop-robber game. *Theoretical Computer Science*, 399(3):179–190, 2008.

[13] A. X. Jiang, Z. Yin, C. Zhang, M. Tambe, and S. Kraus. Game-theoretic randomization for security patrolling with dynamic execution uncertainty. In *AAMAS*, pages 207–214, 2013.

[14] M. Kress, K. Y. Lin, and R. Szechtman. Optimal discrete search with imperfect specificity. *Mathematical methods of operations research*, 68(3):539–549, 2008.

[15] H. Lee, M. Wicke, B. Kusy, O. Gnawali, and L. Guibas. Data stashing: Energy-efficient information delivery to mobile sinks through trajectory prediction. In *ACM/IEEE IPSN*, 2010.

[16] B. Liu, O. Dousse, P. Nain, and D. Towsley. Dynamic coverage of mobile sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 24(2):301–311, 2013.

[17] F. Murtagh and P. Contreras. Algorithms for hierarchical clustering: an overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(1):86–97, 2012.

[18] T. D. Parsons. Pursuit-evasion in a graph. In *Theory and applications of graphs*, pages 426–441. Springer, 1978.

[19] I. Rhee, A. Warrier, M. Aia, J. Min, and M. L. Sichitiu. Z-mac: a hybrid mac for wireless sensor networks. *IEEE/ACM Transactions on Networking (TON)*, 16(3):511–524, 2008.

[20] H. Sato and J. O. Royset. Path optimization for the resource-constrained searcher. *Naval Research Logistics (NRL)*, 57(5):422–440, 2010.

[21] Y.-C. Tseng, Y.-C. Wang, K.-Y. Cheng, and Y.-Y. Hsieh. imouse: an integrated mobile surveillance and wireless sensor system. *Computer*, 40(6):60–66, 2007.

[22] C. Wang, D. De, and W.-Z. Song. Trajectory mining from anonymous binary motion sensors in smart environment. *Knowledge-Based Systems*, 37:346–356, 2013.

[23] D. Wu, S. Ci, H. Luo, Y. Ye, and H. Wang. Video surveillance over wireless sensor and actuator networks using active cameras. *Automatic Control, IEEE Transactions on*, 56(10):2467–2472, 2011.

[24] G. Xing, J. Wang, K. Shen, Q. Huang, X. Jia, and H. C. So. Mobility-assisted spatiotemporal detection in wireless sensor networks. In *IEEE Distributed Computing Systems (ICDCS)*, pages 103–110. IEEE, 2008.

[25] J. Yang, Y. Ge, H. Xiong, Y. Chen, and H. Liu. Performing joint learning for passive intrusion detection in pervasive wireless environments. In *IEEE INFOCOM*, pages 1–9, 2010.

[26] Z. Yin, A. X. Jiang, M. Tambe, C. Kiekintveld, K. Leyton-Brown, T. Sandholm, and J. P. Sullivan. Trusts: Scheduling randomized patrols for fare inspection in transit systems using game theory. *AI Magazine*, 33(4):59, 2012.

[27] Y. Zou and K. Chakrabarty. Sensor deployment and target localization based on virtual forces. In *IEEE INFOCOM*, pages 1293–1303, 2003.