Virtual Scanning Algorithm for Road Network Surveillance

Jaehoon Jeong, *Student Member*, *IEEE*, Yu Gu, *Student Member*, *IEEE*, Tian He, *Member*, *IEEE*, and David H.C. Du, *Fellow*, *IEEE*

Abstract—This paper proposes a *VIrtual Scanning Algorithm* (*VISA*), tailored and optimized for road network surveillance. Our design uniquely leverages upon the facts that 1) the movement of targets (e.g., vehicles) is confined within roadways and 2) the road network maps are normally known. We guarantee the detection of moving targets before they reach designated protection points (such as temporary base camps), while maximizing the lifetime of the sensor network. The main idea of this work is *virtual scan*—waves of sensing activities scheduled for road network protection. We provide design-space analysis on the performance of virtual scan in terms of lifetime and average detection delay. Importantly, to our knowledge, this is the first work to study how to *guarantee* target detection while sensor network deteriorates, using a novel hole handling technique. Through theoretical analysis and extensive simulation, it is shown that a surveillance system, using our design, sustains orders-of-magnitude longer lifetime than full coverage algorithms, and as much as 10 times longer than legacy duty cycling algorithms.

Index Terms—Sensor network, road network, virtual scanning, surveillance, detection, protection.

1 Introduction

Surveillance for critical infrastructure and areas is regarded as one of the most practical applications of wireless sensor networks (WSNs). So far, most of the WSN surveillance systems have focused on surveillance for two-dimensional spaces, such as open battlefields [1], [2], [3], [4]. Research on road network surveillance, however, is very limited. In modern warfare, roadways (as fast maneuver paths) are vantage areas for military surveillance and operations. Clearly, surveillance in a road network is significantly different because 1) the movement of targets (e.g., vehicles) is confined within road segments and 2) the road network maps are normally known (e.g., from Google Earth and Yahoo Maps). We argue that legacy solutions, which are not tailored for road networks, lead to sub-optimal performance.

This paper proposes a novel sensing scheduling algorithm for target intrusion detection, utilizing the unique features of road networks. Specifically, we focus on supporting military operations with fast, infrastructure-free deployment. As shown in Fig. 1a, we guarantee the detection of targets, entering from *entrance points*, before they reach one of the *protection points*; in modern warfare, battlefield situational awareness requires both entrance points and protections points (e.g., temporary base camps) to be assigned and changed on demand for fast military maneuver within a road network. Therefore, we cannot place sensor gates a priori before protection points for intrusion detection. Instead, a road-network-wide deployment is needed.

• *The authors are with the Department of Computer Science and Engineering, University of Minnesota, Twin Cities, EE/CS Building, 200 Union Street SE, Minneapolis, MN 55455.*
E-mail: {jjeong, yugu, tianhe, du}@cs.umn.edu.

Manuscript received 23 Apr. 2009; revised 17 Sept. 2009; accepted 5 Oct. 2009; published online 31 Mar. 2010.
Recommended for acceptance by S. Dal.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-2009-04-0175.
Digital Object Identifier no. 10.1109/TPDS.2010.60.

A straightforward solution for road network surveillance is *duty cycling* in which nodes wake up simultaneously for w seconds (the minimum working time before reliable detection can be reported) and then the whole network remains silent for T seconds. The detection is guaranteed if it takes more than T seconds for a target to travel along the shortest path between any pair of entrance points and protection points; this duty-cycling-based algorithm performs much better in terms of system lifetime than traditional full coverage algorithms [1], [2], [3], [4] in road networks. This is because the duty cycling algorithm allows the whole network to be silent completely for T seconds every w seconds, but the full coverage algorithms (e.g., the one covers all intersections) require at least one subset of sensors to be active at any given point time, taking no advantage of the linear structure of road networks.

In this paper, we present a novel scan-based algorithm, which improves further energy efficiency of surveillance in road networks. As shown in Fig. 1b, sensors wake up one by one for w seconds along road segments, creating waves of sensing activities, called *virtual scanning*. Waves propagate from one (or multiple) protection point P, split at the intersections, and merge along the route until they scan all of the road segments under surveillance. Our study reveals that this scan-based method can achieve significantly better performance (e.g., 10 times system lifetime) than duty cycling algorithms. The concept of virtual scanning is simple; however, in-depth design is very challenging due to a set of practical issues we consider in this paper. Particularly, we investigate 1) how to optimize the network-wide silent duration T between scan waves, 2) how to coordinate the working schedules of individual sensors during the scan, and 3) how to deal with sensing holes due to unbalanced initial node deployment, node failure, and the depletion of node energy over time. Specifically, the intellectual contributions in this paper are as follows:

- A new architecture for surveillance in road networks. *VISA* is the first work tailored for road networks,

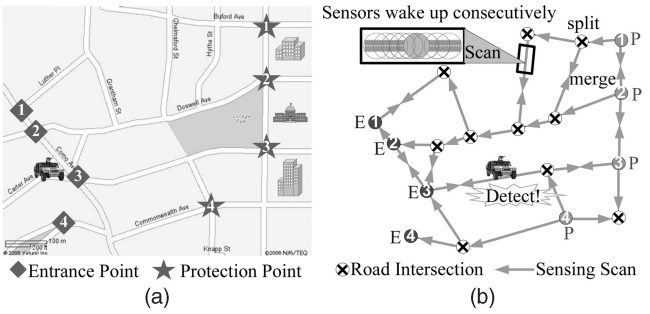

Fig. 1. Road network surveillance. (a) Protected road network. (b) Concept of virtual scanning.

leading to orders-of-magnitude longer system life for target intrusion detection, using a novel scan-based algorithm.

- A sensing scheduling algorithm for an arbitrary road network. The working schedule of each sensor (i.e., when to wake up) is constructed in a decentralized way. The network-wide silent duration is computed by *VISA* scheduler and naturally disseminated along with sensing waves to the nodes in a network.
- An optimal sensing hole handling algorithm for uncovered road segments. The *VISA* scheduler deals with both the initial sensing holes at the deployment time as well as the sensing holes due to the heterogeneous energy budget among sensors by optimally labeling additional pseudoprotection or pseudoentrance points.
- Considerations on two practical issues: 1) Detection failure probability and 2) Time synchronization error. For each issue, we propose an optimal solution in terms of network lifetime.

The rest of the paper is organized as follows: Section 2 describes the problem formulation. Section 3 explains the *VISA* system design. Section 4 evaluates our algorithm through simulation. We summarize related work in Section 5, and finally, conclude the paper with future work in Section 6.

2 Problem Formulation

The goal in this paper is to choose each sensor's sensing schedule in order to maximize the lifetime of a sensor network, while ensuring that all intruding targets are detected before they reach protection points. For clarity, this section explains the basic idea of virtual scanning, using one road segment, and then we extend our design to arbitrary road networks in Section 3.

2.1 Virtual Scanning for Surveillance

We assume that n sensors are randomly placed on a road segment of length l. Each sensor has a conservative sensing circle of radius r, which is long enough to cover the width of the road. This assumption holds true for most commercially available sensors (e.g., PIR sensors can detect moving car 60-100 feet away). Therefore, we can represent sensing coverage using a linear sensor network model as shown in Fig. 2, where n sensors are linearly placed. At the moment, let the left end of the road segment be the entrance point E of targets and the right end of the road segment be the protection point P.

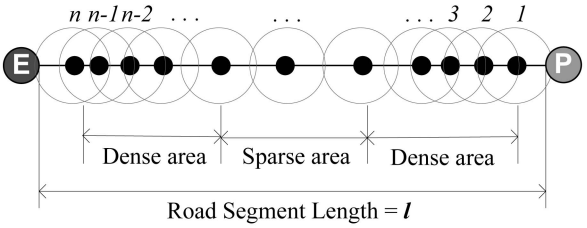

Fig. 2. Randomized linear deployment.

Let w be the minimum working time needed by a sensor in order that the sensor can reliably detect a target over multiple samplings. Let v be a maximum target speed. Suppose that targets enter only from the entrance point and move toward the protection point. In this scenario, we can use the traditional full coverage algorithms, where sensors turn on all the time. We call this approach the ***Always-Awake***.

A better design can be built based on the observation that it takes at least l/v seconds for a target to pass a road segment of length l at a maximum speed v. Therefore, all sensors in the road segment can sleep *together* for l/v seconds, which is defined as *silent time* of the road network. After this *silent time*, all nodes wake up simultaneously for detection. We call this approach ***Duty Cycling***.

Based on the fact that targets move only along the roadways, we propose a new design called ***Virtual Scanning***. As shown in Fig. 3, after all sensors sleep for l/v seconds, we turn on sensors one by one for working time w from the rightmost sensor s_1 toward the leftmost one s_n. Clearly, this wave of sensing activities guarantees the detection and allows *additional sleeping time* for individual sensors. Compared with *Duty Cycling*, this additional sleeping time is obtained by the fact that *all sensors but one can sleep during the scan*. We note that the direction of a virtual scan shall be from the protection point to the entrance point. The virtual scan of the opposite direction (i.e., from the entrance point to the protection point) cannot guarantee target intrusion detection, if a very fast target enters right after the beginning of the network-wide silent time.

2.2 Analytical Network Lifetime Comparison

To understand the key design parameters, this section compares analytically the network lifetime among the

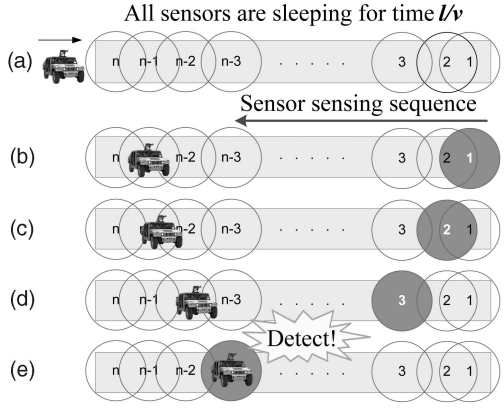

Fig. 3. Sensor sensing sequence.

TABLE 1
Notation of Parameters for Analysis

Parameter	Definition
T_{life}	Lifetime that a sensor can work continuously corresponding to its energy budget.
T_{net}	Sensor network lifetime.
T_{work}	Working time that a sensor needs to work for reliable detection. Normally $T_{work} = w$.
T_{sleep}	Sleeping time of each sensor.
T_{scan}	Scan time that a virtual scan wave moves along the road segment. $T_{scan} = nw$.
T_{silent}	Silent time that the whole sensor network remains silent; that is, time that a target passes through the road segment of length l. $T_{silent} = l/v$.
T_{period}	Schedule period of the sensor network. $T_{period} = T_{scan} + T_{silent}$.

Always-Awake, *Duty Cycling*, and *Virtual Scanning* methods. For clarity, we summarize the notation in Table 1 and overall analytical results in Table 2.

Always-awake and duty cycling. For the *Always-Awake* approach, the network lifetime T_{net} is the same as T_{life} because sensors work continuously without sleeping. For the *Duty Cycling* approach, the network lifetime T_{net} is the number of periods $\lfloor \frac{T_{life}}{w} \rfloor$ multiplied by the length of the period T_{period} (i.e., the sum of the silent time $\frac{l}{v}$ and the working time w):

$$T_{net} = \left\lfloor \frac{T_{life}}{w} \right\rfloor \left(\frac{l}{v} + w \right). \quad (1)$$

Virtual scanning. In the *Virtual Scanning*, the network lifetime T_{net} is the number of periods $\lfloor \frac{T_{life}}{w} \rfloor$ multiplied by the period length T_{period}. T_{period} is the sum of the scan time nw and silent time $\frac{l}{v}$ as shown in Fig. 4. Therefore, we have

$$T_{net} = \left\lfloor \frac{T_{life}}{w} \right\rfloor (T_{scan} + T_{silent}) = \left\lfloor \frac{T_{life}}{w} \right\rfloor \left(nw + \frac{l}{v} \right). \quad (2)$$

Fig. 5 shows the comparison of lifetime among these three approaches. For example, for $w = 1$ sec, *Virtual Scanning* has the lifetime of 30 hours, *Duty Cycling* 3.2 hours, and *Always-Awake* 0.14 hour; *Virtual Scanning* has 9.4 times lifetime of *Duty Cycling* and 214 times lifetime of Always-Awake.

2.3 Analytical Detection Time Comparison

This section compares the average detection time after a target entering a road segment among the *Always-Awake*, *Duty Cycling*, and *Virtual Scanning* methods.

Always-awake and duty cycling. For *Always-Awake*, since a target is detected as soon as it enters the road segment, the average detection time is zero. For *Duty Cycling*, if a target enters during the working period, detection time is zero. On the other hand, if a target enters during the silent time, the average detection time is half of the silent time $l/(2v)$. The percentage of silent time within a period is $l/(wv + l)$, therefore, the overall average detection time of the *Duty Cycling* approach is $l^2/(2v(wv + l))$.

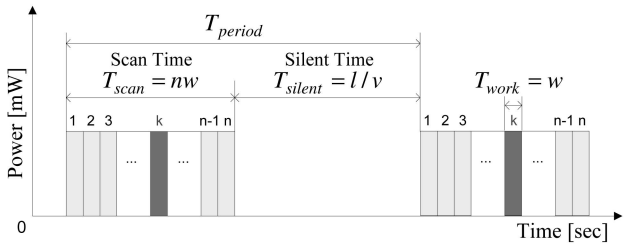

Fig. 4. Scheduling time diagram for node k.

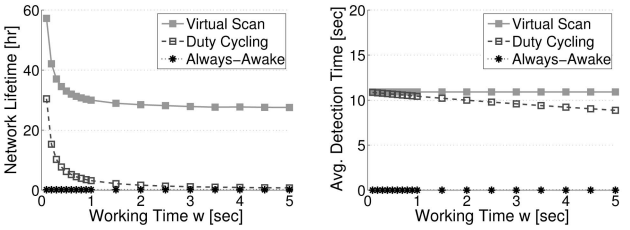

Fig. 5. Performance comparison according to working time.

Virtual scanning. We suppose that n sensors are deployed on a road segment, so each sensor covers the length of l/n in average. Also, we suppose that target speed is v and a target can arrive at any time, that is, the arrival time is uniformly distributed. A target can arrive either during *scan time* or *silent time*. We analyze separately the average detection time for each period and then combine them to obtain overall expected delay $l/(2v)$. Refer to Appendix A the for detailed derivation; note that the average detection time for bounded variable target speed is also derived in Appendix B.

Fig. 5 shows the comparison of average detection time among the three approaches. *Virtual Scanning* detects with a constant delay $l/(2v)$ regardless of working time w. On the other hand, the average detection time of the *Duty Cycling* tends to decrease slowly while working time w increases. The *Always-Awake* method detects without any delay. For example, for working time $w = 0.1$ sec, *Virtual Scanning* has similar performance with *Duty Cycling*, about 10.9 sec. For working time $w = 5$ sec, the *Virtual Scanning* detects target within 10.9 sec, on average, and the *Duty Cycling* does within 8.87 sec; the average detection delay ratio between the *Virtual Scanning* and the *Duty Cycling* is 1.23. However, the ratio of the *Virtual Scanning*'s network lifetime to the *Duty Cycling*'s network lifetime is 37, as shown in Fig. 5. Thus, even though the average detection time increases slightly with *Virtual Scanning*, the benefit of network lifetime is quite remarkable.

TABLE 2
Performance Analysis for Three Approaches

Approach	Sleeping (T_{sleep})	Working (T_{work})	Network Lifetime (T_{net})	Avg. Detection Time
Always-Awake	0	T_{life}	T_{life}	0
Duty Cycling	$\frac{l}{v}$	w	$\lfloor \frac{T_{life}}{w} \rfloor (w + \frac{l}{v})$	$\frac{l^2}{2v(wv+l)}$
Virtual Scanning	$(n-1)w + \frac{l}{v}$	w	$\lfloor \frac{T_{life}}{w} \rfloor (nw + \frac{l}{v})$	$\frac{l}{2v}$

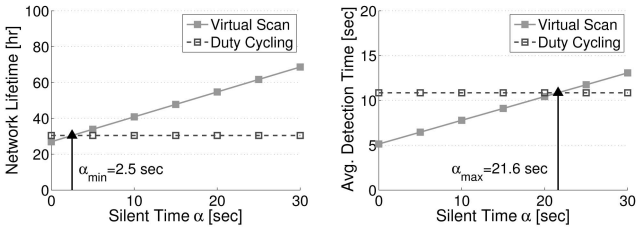

Fig. 6. Performance comparison under different α values.

2.4 Configuring VISA for Better Delay and Longer Lifetime

As a reminder, when the network silent time T_{silent} is equal to or smaller than l/v, target detection is guaranteed. Basic VISA design uses l/v as the network silent time T_{silent}. However, if a smaller silent time T_{silent} is used, it is possible to detect the target not only *faster* but also with *less* energy than the *Duty Cycling* algorithm.

Let $T_{silent} = \alpha$ for $\alpha \in [0, l/v]$. In order to outperform *Duty Cycling* in both network lifetime and average detection delay, we shall satisfy the following inequalities:

$$\begin{array}{lll} \textbf{Virtual Scanning} & & \textbf{Duty Cycling} \\ \left\lfloor \frac{T_{life}}{w} \right\rfloor (nw+\alpha) & \geq & \left\lfloor \frac{T_{life}}{w} \right\rfloor (w+\frac{l}{v}), \\ \frac{l(nw+\alpha)}{2(nwv+l)} & \leq & \frac{l^2}{2v(wv+l)}. \end{array} \quad (3)$$

Solving the above inequalities, we have

$$\max\left\{\frac{l}{v}-(n-1)w, 0\right\} \leq \alpha \leq \min\left\{\frac{l(nwv+l)}{v(wv+l)}-nw, \frac{l}{v}\right\}.$$

When α falls into this range, *Virtual Scanning* has better performance than *Duty Cycling* in both the average detection time and network lifetime. For example, as shown in Fig. 6, for $w = 0.1$ sec, when α is less than $\alpha_{max} = 21.6$ sec, the average detection time of *Virtual Scanning* is shorter than that of the *Duty Cycling*. Also, when α is greater than $\alpha_{min} = 2.5$ sec, the *Virtual Scanning*'s lifetime is longer than *Duty Cycling*'s lifetime. Thus, the range of α achieving better detection delay and lifetime is [2.5, 21.6] sec. We note that the results here only illustrate the idea. Detailed study on the performance effect of α is presented in evaluation Section 4.3.

3 Virtual Scanning Algorithm Design

For the sake of clarity, the previous section presents the basic idea using one road segment. In the rest of the paper, we demonstrate how to apply the virtual scanning to road networks with arbitrary topology. This section is organized as follows: Section 3.1 lists definitions and assumptions used in VISA. Section 3.2 describes the scheduling algorithm and Section 3.3 presents the hole handling algorithm. As practical considerations, Sections 3.4 and 3.5 discuss the handling of dectection failure probability and the handling of time synchronization error, respectively.

3.1 Definitions and Assumptions

Definition 3.1 (Road Network Graph). *Let* Road Network Graph *be* $G = (V, E)$, *where* $V = \{v_1, v_2, \ldots, v_n\}$ *is a set of intersections, entrance points, and protection points in the road network under surveillance, and* $E = [e_{ij}]$ *is a matrix of road segment length* e_{ij} *for vertices* v_i *and* v_j. *Fig. 7 shows a graph* G *corresponding to the road network in Fig. 1. Note that the entrance point set and the protection point set are not static. These two sets can be changed either on demand for military maneuvers or to deal with sensing holes in Section 3.3.*

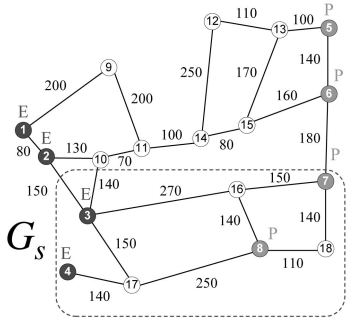

Fig. 7. Road network graph G.

Definition 3.2 (Network Lifetime). *Let* Network Lifetime *be the duration from the starting of a sensor network for surveillance until a* target *can* possibly *reach one of the protection points without detection. In other words, lifetime ends when there exists a possible breach path between an entrance point and a protection point. Note that for the* target, *we cannot differentiate between enemy vehicles and our vehicles from detection since binary sensors are used.*

Definition 3.3 (VISA Scheduler). *Let* VISA Scheduler *be a sink node that initiates the sensing scheduling algorithm.*

The VISA design is based on the following assumptions:

- Road map and locations of sensor nodes are known to *VISA Scheduler*. The sensor location can be obtained through localization schemes [5].
- Sensors are roughly time-synchronized at tens of millisecond level. It can be easily achieved because existing solutions [6], [7] can achieve microsecond-level accuracy.
- Sensors only have simple sensing devices for binary target detection, such as PIR sensors [8]. No sophisticated hardware is available.
- The circular sensing model is used in the conservative way using the minimum sensing range. For irregular sensing modeling, SAM [9] can be used to explore in-situ sensing irregularity.
- One of the existing low-duty-cycle data forwarding schemes, such as DSF [10] and DESS [11], is used to deliver nodes' locations and target detection results to the VISA scheduler.
- Targets move only along the predefined roads with the bounded maximum speed.

3.2 VISA Scheduling on Road Network

This section presents the design of virtual scanning, including schedule establishment and dissemination.

3.2.1 Establishment of Working Schedule

For the clarity of presentation, we use the subgraph G_s of the graph G shown in Fig. 7, where the edge weight means

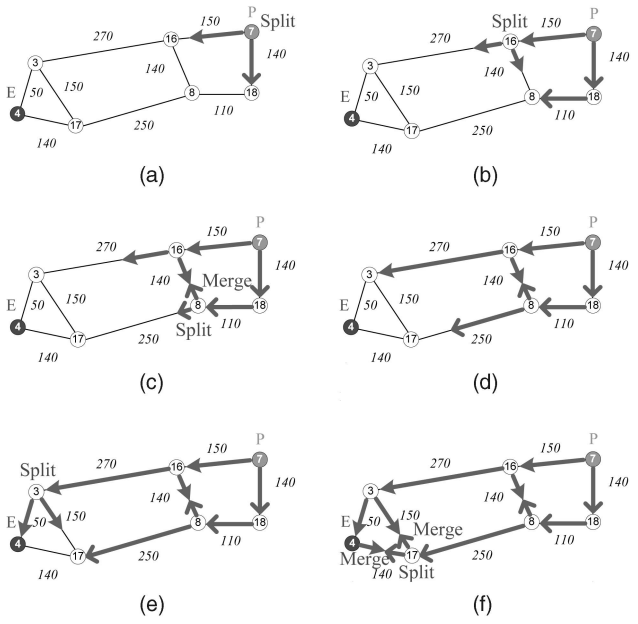

Fig. 8. Virtual scanning on road network for working schedule establishment. (a) Virtual scan originates from protection point v_7 and splits into two scans. (b) Virtual scan arriving at v_{16} splits into two scans toward v_3 and v_8. (c) Two scans from v_8 and v_{16} merge at the road segment (v_8, v_{16}) and stop. (d) Two scans keep going toward intersections v_3 and v_{17}, respectively. (e) Virtual scan arriving at v_3 splits into two scans toward v_4 and v_{17}. (f) Four scans merge at road segments (v_3, v_{17}) and (v_4, v_{17}), respectively.

the physical distance of the road segment. First, we will consider a road network with one entrance and one protection point, and then will consider a road network with multiple entrances and multiple protection points. Also, for now, we assume that no sensing holes exist in the middle of roadways, where targets cannot be detected due to the nonexistence of sensors. The sensing hole handling will be discussed in Section 3.3.

Fig. 8 shows the snapshots of virtual scanning in the road network G_s with one entrance v_4 labeled as E and one protection point v_7 labeled as P; note that one edge $e_{3,4}$ of length 50 is added to G_s in order to explain the virtual scanning. The virtual scan's propagation time on each road segment is the multiplication of the number of sensors and the individual working time w, instead of the physical distance of a road segment. As shown in Fig. 8, by turning on sensors along roads consecutively, virtual scanning waves propagate along multiple routes simultaneously, split at intersections, and disappear when two waves encounter each other in a road segment. Note that once a virtual scanning wave arrives at entrance E, it keeps propagating to the rest of the road network. This scanning guarantees the detection of all of the mobile targets in the road network.

In the case of multiple entrance and protection points, scan operation is similar, except that multiple protection points initiate scanning at the same time. Because the waves merge in the middle of road segments during virtual scanning (as shown in Fig. 8c), regardless of the number of protection points and the locations of the sensors, each sensor only works for w seconds per scan, which is a nice feature for energy balance. Clearly, the scan wave arrival time for each sensor can be easily computed with All-Pairs Shortest Path

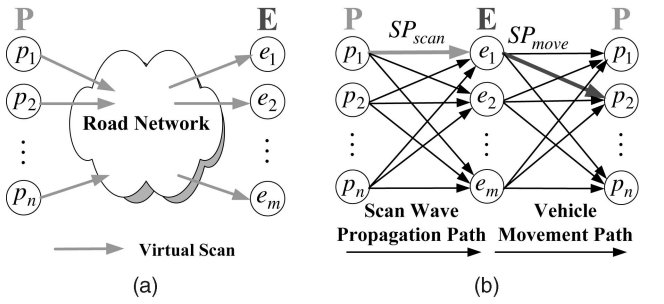

Fig. 9. Virtual scanning on road networks. (a) Virtual scanning on an arbitrary road network with protection points and entrance points. (b) Sleeping time computation considering the shortest scanning and movement paths.

algorithm, such as *Floyd-Warshall algorithm* [12]. We note that the scan wave arrival time decides the working schedule of a sensor node. In other words, a sensor shall start to work for w seconds after a virtual scanning wave arrives.

3.2.2 Establishment of Sleeping Schedule

The previous section discussed how to decide working schedule during the scan. This section explains how to compute the optimal sleeping length, i.e., the maximum duration sensors can sleep safely after working for w seconds while guaranteeing the detection.

Fig. 9a shows the virtual scanning in an arbitrary road network. Let $P = \{p_1, \ldots, p_n\}$ be the set of protection points. Let $E = \{e_1, \ldots, e_m\}$ be the set of entrance points. As discussed before, a period T_{period} consists of 1) silent time T_{silent} during which the whole network is turned off and 2) scan time T_{scan} during which scan waves propagate across the network. Since a sensor only works for fixed $T_{work} = w$ seconds every T_{period}, the longer T_{period} is, the better energy efficiency we have. Therefore, we shall identify the maximum T_{period} value that can guarantee the detection. Before this optimization, we define two important concepts as follows:

Definition 3.4 (The Shortest Scanning Path). *The Shortest Scanning Path $p_{scan}(i,j)$ is the shortest delay path for wave propagation from v_i to v_j on the graph G, where $v_i \in P$ and $v_j \in E$. Let $l_{scan}(i,j)$ be the number of sensors along the path $p_{scan}(i,j)$. Therefore, the Shortest Scanning Time $T_{scan}(i,j)$ can be computed as $l_{scan}(i,j) * w$.*

Definition 3.5 (The Shortest Movement Path). *The Shortest Movement Path $p_{move}(i,j)$ is the shortest distance path between vertices v_i and v_j on the graph G, where $v_i \in E$ and $v_j \in P$. Let $l_{move}(i,j)$ be the shortest distance of $p_{move}(i,j)$. Therefore, the Shortest Movement Time $T_{silent}(i,j)$ can be computed as $l_{move}(i,j)/v_{max}$, where v_{max} is the maximum target speed. We note that all of the sensors along the path $p_{move}(i,j)$ can sleep together for the silent time $T_{silent}(i,j)$.*

These two shortest paths $p_{scan}(i,j)$ and $p_{move}(i,j)$ for all pairs of vertices can be computed based on G by the All-Pairs Shortest Path algorithm, such as *Floyd-Warshall algorithm*.

An important principle of computing the sleeping time is that all of vehicles entering during the sleeping time must be detected before their arrival to the protection points. Once a virtual scan wave originating from the protection points has swept an entrance point, the paths from this

swept entrance point to the protection points are vulnerable to the target intrusion. This is because the swept paths are not swept again until the next scan period.

It is noted that we can guarantee detection by setting T_{period} as the sum of **all-pair minimum scanning time** and **all-pair minimum movement time**. However, the resulting T_{period} is shorter than the optimal value because 1) an intruding target could have to travel a *longer* route from an entrance point with the *earliest* scan arriving time than the shortest route with **all-pair minimum movement time** or 2) it could have to wait until a *late* scan arrives before it can travel along the *shortest* route with **all-pair minimum movement time**, especially when the sensors are nonuniformly placed across a network.

We claim that a longer safe period T_{period} can be obtained as the **minimum sum** of the **scanning time** from v_i to v_j and the **vehicle movement time** from v_j to v_k, for $v_i, v_k \in P$ and $v_j \in E$ than the sum of **all-pair minimum scanning time** and **all-pair minimum movement time**. In order to prove this claim, a three-column graph is introduced as shown in Fig. 9b. The edges between the first and second columns denote the time for wave propagation and the edges between the second and third columns denote the time for target movement. To compute a safe period T_{period}, we need to identify the shortest path from any vertex in the first column to any vertex in the third column. Without loss of generality, suppose that $p_1 \Rightarrow e_1 \Rightarrow p_2$ is the shortest path, where $p_1 \Rightarrow e_1$ is the shortest scanning path from p_1 to e_1, $e_1 \Rightarrow p_2$ is the shortest movement path from e_1 to p_2, and the sum of the scanning time and the movement time for these two paths is the minimum among all of the possible ones. Once the virtual scan arrives at the entrance point e_1 with a delay of $T_{scan}(p_1, e_1)$, the path from the entrance point e_1 to the protection point p_2 becomes vulnerable, if the network remains silent for more than $T_{silent}(e_1, p_2)$. Thus, to prevent a target from reaching the protection point p_2 without detection, another scan wave must be generated from the protection point p_2 after $T_{silent}(e_1, p_2)$. Thus, the safe period is $T_{period} = T_{scan}(p_1, e_1) + T_{silent}(e_1, p_2)$. Note that $p_1 \Rightarrow e_1$ and $e_1 \Rightarrow p_2$ are not necessarily all-pair minimum scanning path and all-pair minimum movement path simultaneously. Therefore, our claim is proved. Consequently, the sleeping time is $T_{sleep} = T_{period} - T_{work}$ because each sensor must work for its duty cycle $T_{work} = w$ seconds per period.

Now, we can formally define the optimization problem of the sleeping time. Suppose that sensors are randomly deployed into the target road network with a uniform sensor density; note that since the number of sensors is proportional to the road segment length, the shortest scanning path is the same as the reverse of the shortest movement path. Let $T_{sleep}(i, j, k) = T_{scan}(i, j) + T_{silent}(j, k) - T_{work}$ for $v_i, v_k \in P$ and $v_j \in E$, where $T_{work} = w$. The optimal sleeping time is chosen as follows:

$$T_{sleep} \leftarrow \min_{\substack{v_i, v_k \in P, \\ v_j \in E}} T_{sleep}(i, j, k). \tag{4}$$

Obviously, the searching for an optimal sleeping time is done in polynomial time $O(mn^2)$. It is noted that only under a uniform sensor density, the formulation in (4) guarantees the optimal sleeping time. Even for a nonuniform sensor density, this formulation provides a good sleeping time close to the optimal one. We leave the optimization for this nonuniform sensor density as future work.

Once the sleeping time value is computed by *VISA scheduler*, it piggybacks in the counter message discussed in Section 3.2.3 and is disseminated to all the sensors in the network. If the VISA scheduler changes the locations of protection and entrance points dynamically, it only needs to recalculate a new sleeping time and redisseminate it.

3.2.3 Decentralized Implementation

In a centralized implementation, a VISA scheduler calculates the working schedules for all the sensors and disseminate the results, which leads to far more messages than necessary ones. Actually, the scan wave arrival time for each sensor can be calculated in a decentralized way. During the initialization phase, all sensors are awake. The sensors at the protection points generate short messages containing a counter with value initialized to one, and pass them to their immediate neighboring sensors. The neighboring sensors only record the minimum counter value ever seen (i.e., discard the rest of messages arriving late), increment the counter, and then relay the message to their neighboring sensors. If a sensor is located at a road intersection, it duplicates and relays multiple copies of messages to all its neighboring nodes except the one it received the message from. In this way, the sensors can decide their sensing scanning order (i.e., the minimum counter value) in the distributed way. Given a sensing order of K, a node shall start to work at time Kw and stop at time $(K+1)w$.

For the sleeping schedule, given the sensor density and the road network along with the entrance and protection points, a conservative sleeping time can initially be estimated by VISA Scheduler; note that sensor density ρ is defined as the number of sensors within the sensing range $2r$, where r is the sensing radius. For example, under a uniform sensor density ρ, we can estimate a conservative sleeping time as *scanning time + movement time − working time* $= \alpha(\frac{l}{2r}\rho w + \frac{l}{v} - w)$, where the shortest path's length is l, the sensing radius is r, the working time is w, the maximum vehicle speed is v, and the coefficient is α (e.g., 0.9). This sleeping time piggybacks in the counter message for the setup of sensors' working schedules in a decentralized way above. After this setup of working schedules, the number of sensors per road segment is reported to VISA Scheduler to determine an optimal sleeping time. This optimal sleeping time will replace the previous conservative sleeping time later.

Up to now, the sensors know when to wake up in order to create virtual scanning (i.e., *Working Schedule* in Section 3.2.1) and how long they can safely sleep with optimal efficiency (i.e., *Sleeping Schedule* in Section 3.2.2). Note that if the entrance point set E and the protection point set P are changed either on demand for military maneuvers or to deal with sensing holes (Section 3.3), both *Working Schedule* and *Sleeping Schedule* are performed for these updated sets.

3.3 Handling of Sensing Holes

We have so far discussed the sensor working schedule and sleeping schedule, assuming balanced energy and no initial sensing holes. In this section, we discuss the handling of sensing holes that can exist after the sensor deployment and that can occur due to the sensor failure or energy depletion. Note that for such sensing holes, we can use sensing hole detection schemes previously studied [13], [14]. As shown

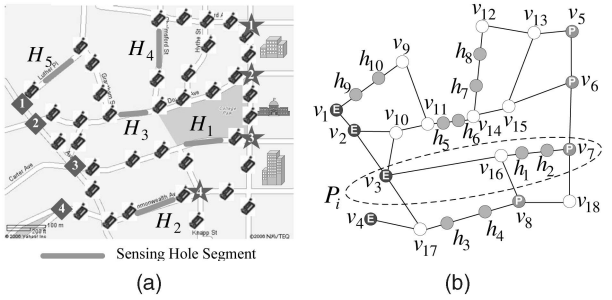

Fig. 10. Augmentation of road network graph for holes. (a) Sensing hole segments on road network: $H_i = (h_j, h_k)$. (b) Augmented graph including sensing holes: $G_a = (V_a, E_a)$.

in Fig. 10a, there exist five sensing hole segments (i.e., $H_1, \ldots, H_5$) that cannot be covered by sensors in the given road network graph. Our idea to deal with these initial hole segments is that we make an augmented graph by adding *the endpoints of the hole segments* (called hole endpoints) as shown in Fig. 10b. Note that these initial hole segments and the corresponding hole endpoints can be identified by *VISA Scheduler* since sensors report their locations to *VISA Scheduler* at the initialization phase. To ensure the protection, we treat the hole endpoints as either pseudoentrance points or pseudoprotection points. The hole handling problem is, therefore, reduced to a labeling problem of hole endpoints.

Problem Definition: How to optimally determine the role of each hole endpoint (i.e., label as either entrance point or protection point) in order to achieve the maximum sleeping time, leading to the maximization of the sensor network lifetime.

In the rest of this section, we present an optimal labeling algorithm for hole handling.

3.3.1 Initial Sensing Holes

In reality, there is high probability that some road segments are not covered by sensors even though many sensors are randomly deployed on road network as shown in Fig. 10a. We define these uncovered road segments as the *initial sensing hole segments*; note that each sensing hole segment consists of two hole endpoints.

Suppose that n hole endpoints occur under a uniform sensor density. With an exhaustive search, 2^n cases are required to investigate. This means the time complexity of $O(2^n)$. Since this complexity is intractable, we need an improved way to achieve an optimal labeling for hole endpoints.

We explain here the idea with a simplified example; Fig. 10b shows one roadway P_i consisting of v_3, v_{16}, and v_7 and a hole segment H_1 with hole endpoints h_1 and h_2, which are closer to a protection point v_7 than an entrance point v_3. If two hole endpoints h_1 and h_2 are labeled differently, this short hole segment determines the shortest sleeping time. To avoid this, h_1 and h_2 should have the same type of label. Furthermore, since h_1 and h_2 are near the protection point v_7, in order to get a longer sleeping time, they should be labeled as protection points.

Conceptually, when labeling hole endpoints, we should label each hole endpoint with the same label as the closest point already labeled. Rationale behind this insight is: *the maximization of the path distance between the entrance points and protection points leads to a maximum sleeping time according to (4).*

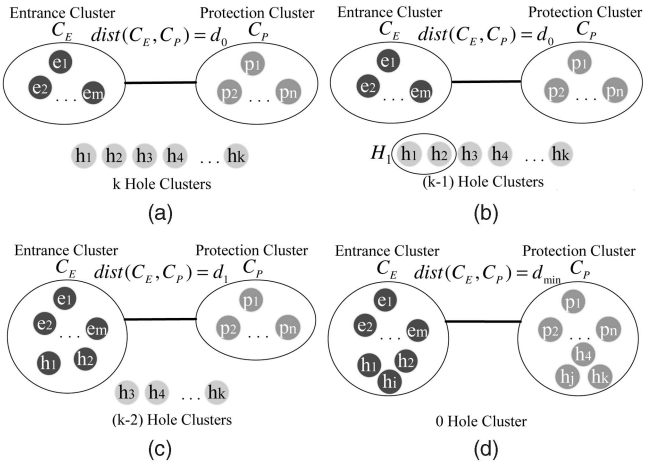

Fig. 11. Clustering for sensing hole labeling. (a) Initial clusters: entrance cluster C_E, protection cluster C_P, and k hole clusters h_s for $s = 1..k$. (b) The closest pair of h_1 and h_2 are merged into cluster H_1 labeled as unknown. (c) The closest pair of C_E and H_1 are merged into C_E with H_1 labeled as entrance label L_E. (d) After clustering, all hole clusters are merged into either C_E or C_P, that is, labeled as either L_E or L_P.

Formally, let H be the set of hole endpoints such that $H = \{h_1, h_2, \ldots, h_k\}$. Let E be the set of entrance points and P be the set of protection points. Let L_E be the entrance label and L_P be the protection label. We can label the holes in H, by partitioning H into two disjoint subsets (called *clusters*) Entrance Cluster (C_E) and Protection Cluster (C_P). Asano et al. proposed such a clustering algorithm for a farthest k-partition based on Minimum Spanning Tree (MST) [15], giving an *optimal clustering* to maximize the intercluster distance. We extend Asano et al.'s *Clustering* for sensing hole labeling.

Fig. 11 illustrates the main idea. Let $dist(C_E, C_P)$ be the intercluster distance between C_E and C_P. Our objective is to partition the set H into two disjoint sets C_E and C_P such that the intercluster distance between C_E and C_P is maximized. The initial intercluster distance is $dist(C_E, C_P) = d_0$, as shown in Fig. 11a. In this example, suppose that two hole clusters h_1 and h_2 are the closest pair of two clusters. In this case, these hole clusters are merged into one hole cluster H_1 with the same, unknown label, as shown in Fig. 11b. The reason two clusters h_1 and h_2 are merged into one hole cluster with the same label is to let the intercluster distance between C_E and C_P be maximized. Otherwise, the intercluster distance between h_1 and h_2 can be the intercluster distance shorter than the initial intercluster distance $dist(C_E, C_P) = d_0$. As shown in Fig. 11c, two clusters C_E and H_1 are the closest pair, so H_1 is merged into C_E with hole endpoints h_1 and h_2 labeled as entrance. In this way, we can cluster all of the hole endpoints into either C_E or C_P to maximize the intercluster distance $dist(C_E, C_P)$, as shown in Fig. 11d. Similar to Asano et al.'s algorithm [15], our clustering gives an optimal hole labeling because it satisfies the *greedy choice property* and *optimal substructure* [12].

As an important difference from Asano et al.'s *Clustering*, during the clustering, we maintain multiple hole clusters H_i labeled as *unknown* in addition to one Entrance Cluster C_E

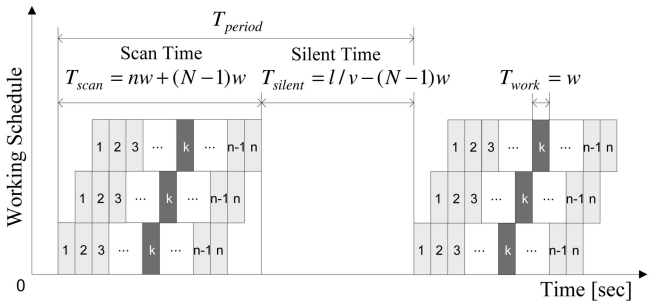

Fig. 12. Scheduling time diagram for redundant sensing.

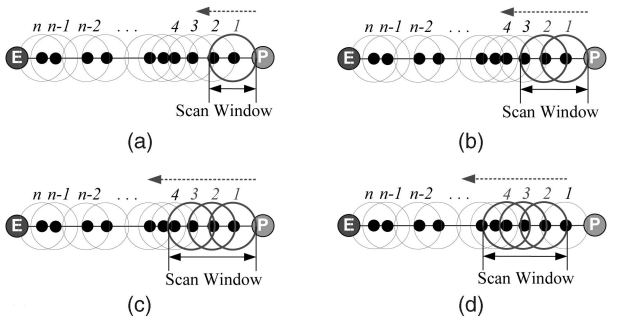

Fig. 13. Virtual scanning for redundant sensing. (a) Scan by sensor 1. (b) Scan by sensors 1 and 2. (c) Scan by sensors 1, 2, and 3. (d) Scan by sensors 2, 3, and 4.

and one Protection Cluster C_P. Through the MST construction, we merge one hole cluster H_i to either C_E or C_P such that the intercluster distance between C_E and C_P is maximized. We call this new labeling algorithm the *MST-based Labeling*.

3.3.2 Sensing Holes due to Energy Depletion or Failure

In the previous section, we discussed the initial sensing hole issue. However, since in reality, sensors deployed on a road network may not have the same amount of energy initially, we need to consider the sensing holes caused by this unbalanced sensor energy budget. Also sensor could fail over time. We can deal with these sensing holes in the same way as with the initial holes; we can either completely relabel all holes or incrementally label new holes by using *MST-based Labeling*. The former is optimal, but the latter introduces less computation.

To detect the failure of sensors due to energy depletion, the sensors can regularly report their existence to *VISA Scheduler*. Also, in a distributed way, the neighboring nodes can exchange their health status with each other in a regular basis. We leave this kind of fault node detection as future work because in *VISA*, the fault node detection is not a key design component.

3.4 Handling of Detection Failure Probability

In reality, there exist sensing errors in sensors. We need to relax the assumption that every vehicle within the sensing range of some sensors can be detected with probability one. Let p be the probability of a success (called *detection success probability*) on each sensing for working time w. When the required network-wide detection probability for mobile targets is given as P_{req} (e.g., 0.99) and the sensor's detection success probability is known as $p = 0.9$, the question is how to schedule sensors in order to achieve the user-required detection probability P_{req}. The idea is to perform multiple sensing activities. For example, let q be the probability of a failure (called *detection failure probability*), where $q = 1 - p$. When $p = 0.9$, $q = 0.1$, which means that there exists one detection missing among 10 detection trials. Let $P_{req} = 0.99$, which means that there exists one detection missing among 100 detection trials. Assume that a mobile target is staying at a sensor's sensing range. In order to achieve $P_{req} = 0.99$, the sensor needs to perform two consecutive sensing activities because the corresponding network-wide detection failure probability is $1 - P_{req} = 0.01$ and the detection failure of two consecutive sensing activities is $(1-q)^2 = 0.01$. Another way is for two sensors to perform their sensing activity simultaneously for the mobile target. The first approach is defined as *temporal redundant sensing* and the second as *spatial redundant sensing*. Our scheme to deal with the detection failure probability is the combination of these two approaches, defined as *redundant sensing*.

First, we explain the number of redundant sensing activities N given q and P_{req}. Let $\bar{P}_{req}$ be the network-wide detection failure probability such that $\bar{P}_{req} = 1 - P_{req}$. Suppose the sensing activities are independent and identically distributed (*i.i.d.*). The number N can be computed as follows:

$$q^N \leq \bar{P}_{req} \Rightarrow N = \lceil \log_q \bar{P}_{req} \rceil. \quad (5)$$

Fig. 12 shows the scheduling time diagram for the *redundant sensing* in the *Virtual Scanning*. As shown in the figure, each sensor performs N sensing activities consecutively. For the *redundant sensing*, we perform N virtual scans consecutively from the protection point toward the entrance point. To guarantee that a mobile target is scanned at least N times, the silent time T_{silent} is reduced to $\frac{l}{v} - (N-1)w$. This is because N scans should be started before the mobile target reaches the protection point.

Fig. 13 shows the virtual scanning for the redundant sensing, where $N = 3$, which is equivalent to the scheduling time diagram in Fig. 12. The scan window is defined as the cluster of adjacent working sensors. Initially, as shown in Fig. 13a, when sensor s_1 is working, the size of the scan window is 1. After the working time w, as shown in Fig. 13b, the size becomes 2 since two sensors s_1 and s_2 are working together. After another w, as shown in Fig. 13c, the size becomes 3 since three sensors s_1, s_2, and s_3 are working together. Finally, after another w, as shown in Fig. 13d, the scan window of size 3 is shifted to the left because the number N of simultaneous working sensors is 3.

For the *Duty Cycling*, each sensor performs N sensing activities per its duty cycle in order to provide the same detection probability as with the *Virtual Scanning*. On the other hand, for the *Always-Awake*, no change is required for the redundant sensing, because it can already provide such a redundant sensing. We summarize the performance analysis for these three approaches in Table 3 in terms of the maximum network lifetime T_{net}. Note that the network lifetime is usually less than T_{net} since a vehicle can pass the sensor network at any time without detection, where each sensor can detect a vehicle with detection success probability p per sensing trial, that is, as the sensor network tries to detect the vehicle N times with the detection success probability p

TABLE 3
Performance Analysis for Three Approaches in Redundant Sensing

Approach	Sleeping (T_{sleep})	Working (T_{work})	Maximum Network Lifetime (T_{net})
Always-Awake	0	T_{life}	T_{life}
Duty Cycling	$\frac{l}{v} - (N-1)w$	Nw	$\lfloor \frac{T_{life}}{Nw} \rfloor (Nw + \frac{l}{v} - (N-1)w)$
Virtual Scanning	$(n-1)w + \frac{l}{v} - (N-1)w$	Nw	$\lfloor \frac{T_{life}}{Nw} \rfloor (nw + \frac{l}{v})$

per sensing trial, it can fail detecting the vehicle with probability $(1-p)^N$ for N trials per duty cycle.

3.5 Handling of Time Synchronization Error

Up to this point, it is assumed that sensors in *VISA system* are roughly time-synchronized as long as there is no time gap between the working schedules of two neighboring sensors during the scan time for vehicle detection. Considering that many state-of-the-art solutions [6], [7] can provide sensors with the time synchronization at the microsecond level, we explain how to perform the virtual scanning to satisfy this assumption in this section.

Fig. 14 illustrates the handling of time synchronization error by the overlap of the working schedules through the margin of sensing time. As shown in Fig. 14a, there exist time gaps among the working schedules of sensors s_{k-1}, s_k, and s_{k+1}. However, as shown in Fig. 14b, through the margin of the working time based on the maximum time synchronization error ϵ_{max}, the working schedules have time overlaps, guaranteeing no time gaps among the working schedules.

This detection guarantee can be explained in a more formal way as follows: Suppose that a maximum time synchronization error is known as ϵ_{max}, sensor s_k is required to have a margin of ϵ_{max} for its working start time t_k^s and working end time t_k^e such that the working schedule is $[t_k^s - \epsilon_{max}, t_k^e + \epsilon_{max}]$. This allows two adjacent sensors' working schedules to overlap even under maximum time synchronization errors. It can be explained that this working schedule with a margin of ϵ_{max} guarantees the target detection as follows: Suppose that the working time is w, the accurate working schedules of two adjacent sensors s_k and s_{k+1} are $[t^*, t^* + w]$ and $[t^* + w, t^* + 2w]$, respectively, and their synchronization errors are ϵ_k and $\epsilon_{k+1} \in [-\epsilon_{max}, \epsilon_{max}]$, respectively. Let us consider the worst scenario for the maximum time gap between these two working schedules, that is, $\epsilon_k = -\epsilon_{max}$ and $\epsilon_{k+1} = \epsilon_{max}$. This setting makes the following working schedules: 1) s_k's working schedule $= [t^* - \epsilon_{max}, t^* + w - \epsilon_{max}]$ and 2) s_{k+1}'s working schedule $= [t^* + w + \epsilon_{max}, t^* + 2w + \epsilon_{max}]$. Now, we augment these two schedules with the maximum time synchronization error ϵ_{max} for the safe detection as follows: 1) s_k's new working schedule $= [t^* - 2\epsilon_{max}, t^* + w]$ and 2) s_{k+1}'s new working schedule $= [t^* + w, t^* + 2w + 2\epsilon_{max}]$. Thus, these new working schedules have no time gap, so the targets can be detected without missing. Note that the sleeping time T_{sleep} for sensor s_k decreases by $2\epsilon_{max}$ since the working time T_{work} becomes $w + 2\epsilon_{max}$ in the schedule period T_{period}. *Duty Cycling* also needs to adjust each sensor's working time and sleeping time by $w + 2\epsilon_{max}$ in the same way as with *Virtual Scanning*. On the other hand, *Always-Awake* does not need this adjustment since it does not have any sleeping time.

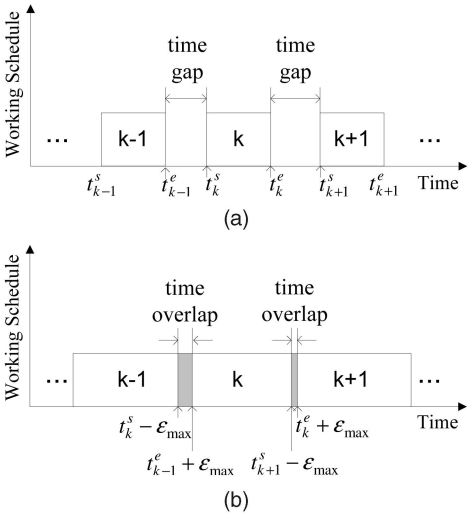

Fig. 14. Handling of time synchronization error. (a) Scheduling time diagram with time gap. (b) Scheduling time diagram with time overlap.

4 Performance Evaluation

In this section, we analyze performance of VISA, comparing with other schemes for road network surveillance.

- **Performance Metrics:** We use *network lifetime* and *average detection time* as the performance metrics.
- **Baselines:** Since the road network surveillance is a new research area, to the best of our knowledge, there exist no other state-of-the-art sensing schemes for road network surveillance. We compare VISA with two approaches: *Duty Cycling* and *Always-Awake*.
- **Parameters:** In the performance comparison, we investigate the impact of the following four parameters:

  1. working time,
  2. sensor density,
  3. time synchronization error, and
  4. detection failure probability.

  In addition, we reveal 1) the impact of sleeping time duration and 2) the effect of sensing hole labeling.

Simulation uses the map of a real road network as shown in Fig. 7. For vehicle mobility, vehicles arrive at the specified entrances of the road network and randomly choose one protection point as destination, moving toward the destination via the shortest path. The vehicle arrival time is uniformly distributed during the system lifetime with mean interarrival time 60 sec. The system parameters are selected based on a typical military scenario [16]. Unless mentioned otherwise, the default values in Table 4 are used. Based on

TABLE 4
Simulation Configuration

Parameter	Description
Sensing range R	$R = 2r = 20$ meters (i.e., 66 feet) where r is sensing radius.
Energy budget b	$b \sim N(\mu_b, \sigma_b)$ where $\mu_b = 50$ kilo-joule (kJ) and $\sigma_b = \{0, 2, ..., 18\}$ kJ. The default of σ_b is 5 kJ.
Sensor working time w	Nine points in [0.1,0.9] with step time 0.1 sec and nine points in [1,5] with step time 0.5 sec. The default of w is 1 sec.
Sensor density d	$d \sim N(\mu_d, \sigma_d)$ where $\mu_d = \{2, 4, ..., 20\}$ and $\sigma_d = \{0, 1, ..., 6\}$. The default of (μ_d, σ_d) is $(10, 1)$.
Detection failure probability q	$q = \{0, 0.05, 0.1, ..., 0.4\}$. The default of q is 0, that is, 100% detection.
Time synchronization error ϵ	$\epsilon \sim Uniform(-\epsilon_{max}, \epsilon_{max})$ where $\epsilon_{max} = \{0, 0.1, ..., 1\}$ sec. The default of ϵ is 0, that is, no time synchronization error.
Vehicle speed v	$v \sim N(\mu_v, \sigma_v)$ where $\mu_v = \{15, 20, ..., 60\}$ MPH and $\sigma_v = \{0, 1, ..., 10\}$ MPH. The maximum speed is 70 MPH and the minimum speed 10 MPH. The default of (μ_v, σ_v) is $(40, 5)$.

these settings, we implemented our own event-driven simulator for evaluation.

For network lifetime measurement, the default energy budget (50 kJ) is used, but for the average detection time measurement, to obtain high statistical confidence, a full-day energy budget is used for the comparison among the three approaches.

4.1 System Behavior over Time

All three methods *Virtual Scanning*, *Duty Cycling*, and *Always-Awake* can guarantee the detection of targets. Their difference lies in the network lifetime. Clearly, as a node can sleep longer per period with detection guarantee, the more energy efficiency can be obtained. Fig. 15 shows how the sleeping time T_{sleep} changes before network lifetime ends. As shown in the figure, *Virtual Scanning* has by far the longest sleeping time, and hence, the longest network lifetime. For example, *Virtual Scanning* sustains for 28.2 hours, compared with 1.4 hours in *Duty Cycling* and 5.4 minutes in *Always-Awake*. This is because of the significant energy savings during the scanning process. Note that in Fig. 15, *Virtual Scanning* lets the sleeping time degrade gradually. This is because as the sensors are dying over time due to energy depletion, the sensing holes occur. These sensing holes let the distance between the entrance cluster and the protection cluster get shorter, as discussed in Section 3.3. Thus, the shorter intercluster distance leads to the shorter sleeping time. In the following sections, we will quantitatively show the effect of the sleeping time on the performance.

4.2 Performance Comparison

In this section, we compare three approaches: 1) *Virtual Scanning*, 2) *Duty Cycling*, and 3) *Always-Awake* in terms of Network Lifetime and Average Detection Time under several user-level parameters, such as working time, sensor density, time synchronization error, and detection failure probability. Each point in each experiment is the mean of the results obtained with 10 different random seeds.

4.2.1 The Impact of Working Time

Since w is the minimum working time before reliable detection can be reported, this evaluation reveals how different hardware response speeds and sensing algorithms affect the VISA and other baselines. We use nonuniform 50 kJ energy budget with the energy variation 5 kJ. Clearly, VISA provides significantly longer system lifetime than the baselines, especially when w is large as shown in Fig. 16a. For example, when w is 1 second, VISA extends network lifetime by 18.5 times compared with *Duty Cycling*, and 146 times compared with *Always-Awake*. As shown in Fig. 16b, the average detection time of *Virtual Scanning* is 11.5 sec, which is two times longer than that of the *Duty Cycling*, 5.8 sec. Therefore, *Virtual Scanning* can provide 19 times lifetime of *Duty Cycling* at the expense of two times longer average detection time.

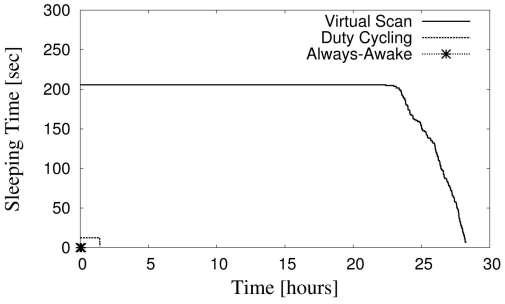

Fig. 15. The comparison of sleeping time T_{sleep} over time.

4.2.2 The Impact of Sensor Density

We define *sensor density* as the average number of sensors within sensing range R. As expected from the formula of the network lifetime in (2), the high sensor density provides the longer network lifetime for *Virtual Scanning*, as shown in Fig. 17a. This is because with a higher density, we have a longer scanning time T_{scan}, which allows sensor nodes to sleep longer. However, the high sensor density does not contribute much to the network lifetime to *Duty Cycling* and *Always-Awake*, since their sleeping time is independent of the number of sensors (as shown in Table 2).

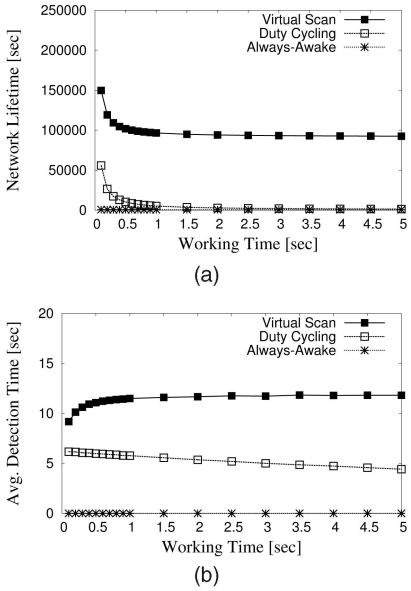

Fig. 16. The impact of working time w. (a) Network lifetime versus working time. (b) Average detection time versus working Time.

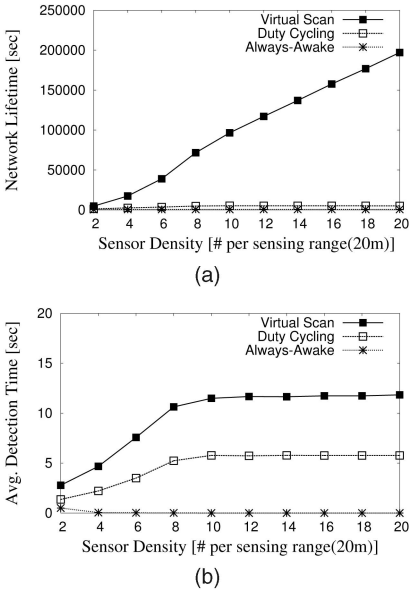

Fig. 17. The impact of sensor density d. (a) Network lifetime versus sensor density. (b) Average detection time versus sensor density.

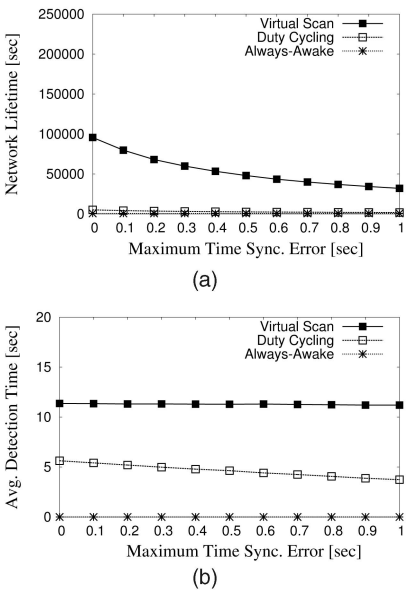

Fig. 18. The impact of time synchronization error ϵ_{max}. (a) Network lifetime versus time synchronization error. (b) Average detection time versus time synchronization error.

For the average detection time, in both *Virtual Scanning* and *Duty Cycling*, under sparse sensor density less than 8, the lower density lets the sensors close to entrances detect vehicles earlier, leading to the shorter average detection time, as shown in Fig. 17b. This is because for initial sensing holes due to low sensor density, the corresponding hole endpoints are labeled as entrance points or protection points for the sensing hole handling discussed in Section 3.3. As the sensor density becomes lower, the distance between the entrance cluster and the protection cluster is getting shorter, leading to the shorter average detection time according to *Average Detection Time* equations in Table 2. On the other hand, under sensor density greater than 8, since the distance between the entrance cluster and the protection cluster is constant regardless of the sensor density, the average detection time is almost the same. In summary, at all sensor density settings, *Virtual Scanning* provides the longest network lifetime with a small degradation in detection time (e.g., double detection latency), compared with the lifetime increase (e.g., 19 times) where sensor density is 10 with $w = 1$ sec; note that this performance gain becomes higher when sensor density becomes higher.

4.2.3 The Impact of Time Synchronization Error

We investigate the impact of the time synchronization error on the network lifetime and average detection time. For this investigation, the margin of working time is set to the maximum time synchronization error ϵ_{max} for the safe detection. As shown in Fig. 18a, as ϵ_{max} increases, the network lifetimes of both *Virtual Scanning* and *Duty Cycling* decrease. This is because each sensor's working time increases by $2\epsilon_{max}$ and its sleeping time decreases by $2\epsilon_{max}$, as discussed in Section 3.5.

For the average detection time, as shown in Fig. 18b, *Virtual Scanning* has almost the same average detection time regardless of ϵ_{max}. This is because the average detection time is independent of the working time changed by ϵ_{max}, as shown in Table 2. On the other hand, the average detection time of *Duty Scanning* tends to decrease as ϵ_{max} increases. This is because the average detection time is inversely proportional to the working time increased by $2\epsilon_{max}$, as shown in Table 2.

4.2.4 The Impact of Detection Failure Probability

In this section, we investigate the impact of detection failure probability q along with the following two parameters: 1) working time w and 2) sensor density d. Fig. 19 shows the impact of the working time on the performance in the redundant sensing to deal with the detection failure probability. In the simulation setting, the number of redundant sensing activities is 3 from (5) in Section 3.4 where q is set to 0.1

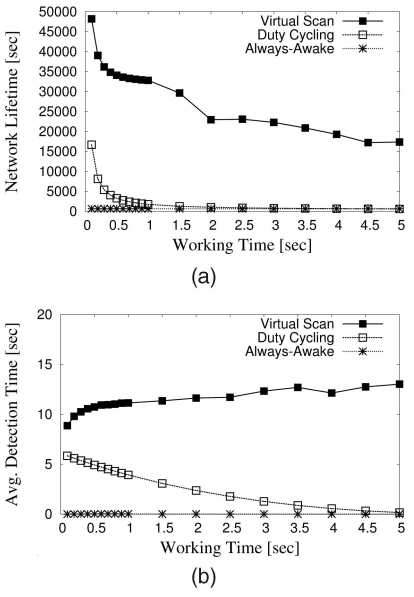

Fig. 19. The impact of working time w under redundant sensing. (a) Network lifetime versus working time. (b) Average detection time versus working time.

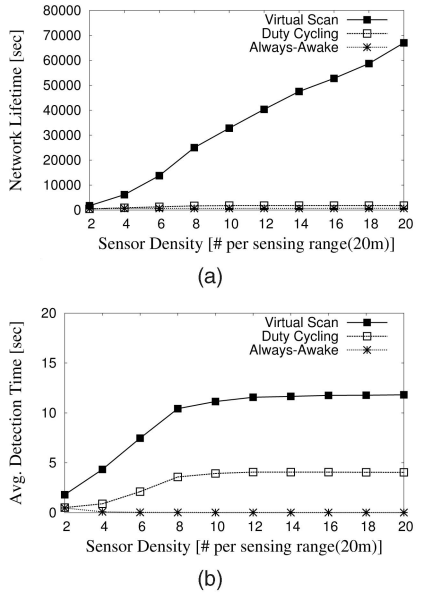

Fig. 20. The impact of sensor density d under redundant sensing. (a) Network lifetime versus sensor density. (b) Average detection time versus sensor density.

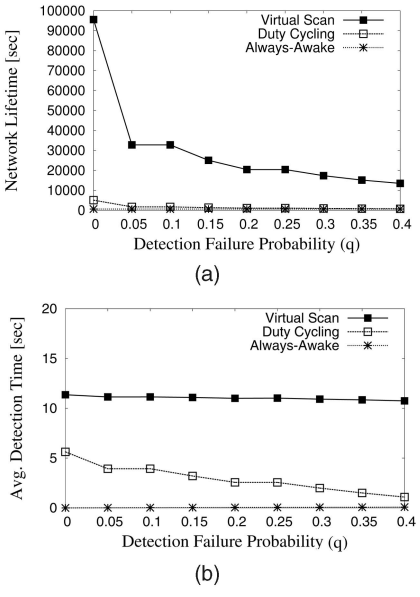

Fig. 21. The impact of detection failure probability q. (a) Network lifetime versus failure probability. (b) Average detection time versus failure probability.

and the required network-wide detection probability is 0.999. That is, since the working time in the redundant sensing is three times longer than that in the nonredundant sensing in Section 4.2.1, the lifetime in the redundant sensing is at most one third of that in the nonredundant sensing; note that this maximum lifetime is possible only when vehicles can be detected until the energy budgets of the sensors are depleted completely.

As shown in Fig. 19a, for *Virtual Scanning*, this expectation is valid when the working time is short, such as from 0.1 to 1 sec; this is caused by the fact that in a short working time, neighboring sensors can perform more than three sensing activities. For example, suppose that two sensors s_1 and s_2 are almost in the same location and the working time w is 0.1 sec. For the sensing circle of 20 meters, a vehicle takes almost 1 second to pass this sensing circle with the speed of 40 MPH. These two sensors can try to detect the vehicle three times, respectively; the total number of trials becomes 6. Thus, the network-wide detection failure probability is $q^6 = 10^{-6}$, which is a very small number. On the other hand, for the working time longer than 1 sec, the lifetime is decreasing because the network-wide detection can fail earlier than the maximum lifetime with only three detection trials; in this case, the network-wide detection failure probability is $q^3 = 10^{-3}$. Thus, the benefit of the spatial redundant sensing decreases as the working time increases. Note that in Fig. 19a, the curve is not smooth from $w = 1$ sec to $w = 2$ sec. After $w = 1$ sec, the network-wide detection failure probability dramatically increases with less spatial redundant sensing. For *Duty Cycling*, the lifetime tends to decrease as the working time increases in the similar way with the case of the nonredundant sensing. For $w = 0.1$ sec, *Virtual Scanning* has 2.89 times longer lifetime than *Duty Cycling*. For the working time longer than 0.5 sec, *Virtual Scanning* has at least 10 times longer lifetime than *Duty Cycling*.

As shown in Fig. 19b, the average detection time of *Virtual Scanning* increases slightly as the working time increases. This is because the network-wide detection failure probability increases with less spatial redundant sensing according to the increase of the working time, leading to a little later detection. For *Duty Cycling*, the average detection time decreases faster for the increase of the working time than that in the nonredundant sensing, as shown in Fig. 19b. This is because the redundant sensing increases the working time per duty cycle; note that the average detection time is inversely proportional to the working time.

For the sensor density, as shown in Fig. 20, the patterns of the curves of *Virtual Scanning* and *Duty Cycling* are similar to those in the nonredundant sensing, as shown in Fig. 17. The remarkable difference in the network lifetime is that the lifetimes of *Virtual Scanning* and *Duty Cycling* are reduced to almost one third. For the average detection time, *Virtual Scanning* has almost the same detection time as with the case of the nonredundant sensing. The average detection time of *Duty Cycling* is shorter than the case of the nonredundant sensing. This is because the average detection time is inversely proportional to the working time and the working time is increased due to the redundant sensing.

Now, we investigate the impact of detection failure probability on both performance metrics. For the lifetime, as shown in Fig. 21a, *Virtual Scanning* has shorter lifetime according to the increase of detection failure probability q, leading to the increasing number of sensing activities per duty cycle. However, the performance gain of *Virtual Scanning* in the lifetime is at least 17 times over *Duty Cycling* and at least 20 times over *Always-Awake*. For the average detection time, as shown in Fig. 21b, *Virtual Scanning* has almost the constant detection time, however *Duty Cycling* tends to have a shorter detection time according to the increase of q; this is because the increase of q leads to the increase of the working time.

In summary, even in the realistic setting with the time synchronization error and detection failure probability, *Virtual Scanning* outperforms both *Duty Cycling* and *Always-Awake* in terms of network lifetime.

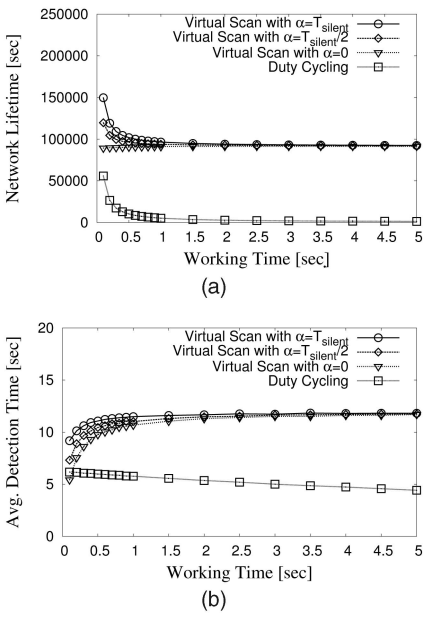

Fig. 22. The impact of silent time α. (a) Network lifetime versus working time. (b) Average detection time versus working time.

4.3 Achieving Shorter Delay and Longer Lifetime Simultaneously

In this section, we show that there exists a working time such that *Virtual Scanning* is better in both the detection delay and the lifetime than *Duty Cycling*. In Section 2.4, we showed analytically how VISA achieves a shorter delay and a longer network lifetime simultaneously by adjusting the *silent time* ($T_{silent} = \alpha$) within the range that satisfies (3), where the silent time is part of the sleeping time.

To confirm our design empirically, Fig. 22 shows the performance effect of *Virtual Scanning* according to α. As shown in Fig. 22, when *Virtual Scanning* reduces α from T_{silent} to 0 in the working time of 0.1 second, it has better performance in both the network lifetime and average detection time than *Duty Cycling*. Therefore, the system operator can achieve the required performance by tuning the working time and the sleeping time.

4.4 The Effect of Hole Handling

This section compares three different methods for hole handling:

- *MST-based Labeling*: our hole labeling scheme discussed in Section 3.3.
- *Random Labeling*: a new hole is randomly labeled as either pseudoentrance point or pseudoprotection point.
- *No Labeling*: when a new hole occurs, it is not handled, leading to the end of the system lifetime.

We use the same *Virtual Scanning* for these three labeling algorithms. As shown in Fig. 23, *MST-based Labeling* gives longer lifetime than both *Random Labeling* and *No Labeling*. *Random Labeling* and *No Labeling* have the similar lifetime because *Random Labeling* cannot label holes appropriately to prevent a breach path (i.e., path vulnerable to vehicle intrusion to protection points) from existing; that is, since *Random Labeling* might label holes against our labeling rule based on MST discussed in Section 3.3.1 (i.e., to label two closest holes with the same label for longer lifetime), the holes close to the entrance points can have different labels, leading to the short sleeping time. Since *No Labeling* does not handle sensing hole, one sensing hole creates a breach path, leading to the end of the system.

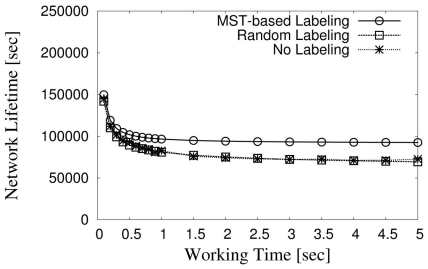

Fig. 23. The comparison of hole labeling algorithms.

For the average detection time, these three labeling algorithms have similar performance whose curves are almost the same as the curve of *Virtual Scanning* in Fig. 16b.

5 Related Work

Most research on coverage for detection has so far focused on *Full Coverage* [1], [2], [3], [4], [17], [18], [19], [20] in a two-dimensional space. In [4], the authors use the off-duty eligibility rule to turn on/off a node as long as the neighboring nodes can cover the sensing area of this node. The Coverage Configuration Protocol (CCP) [18] provides an energy-efficient sensing coverage, integrated with SPAN for connectivity. In [21], surveillance coverage is achieved through probing. DiffSurv [22] provides differentiated surveillance to an area with a certain degree of coverage, up to the limitation imposed by the number of sensor nodes deployed. Compared with this static coverage, Cheng et al. [23] propose sweep coverage based on mobile sensors. These mobile sensors monitor certain points of interest such that these points are swept by the mobile sensors at least once every required period. Cheng et al. also investigate theoretically the minimum number of mobile sensors to satisfy the required sweep coverage. Kumar et al. [3] identify a critical bound for k-coverage in a network, assuming that a node is randomly turned on with a certain probability. Cardei et al. [2] propose two heuristic algorithms to identify a maximum number of set covers to monitor a set of static targets at known locations. Abrams et al. [1] propose three approximation algorithms for a relaxed version of the previously defined SET K-COVER problem [24]. Even for SET K-COVER setting, our *Virtual Scanning* outperforms the *Duty Cycling* in terms of lifetime. For each cover, *Virtual Scanning* has a longer schedule period according to scan time than *Duty Cycling*, leading to the longer lifetime. We leave the application of SET-K-COVER to *Virtual Scanning* as future work.

To aggressively reduce the energy consumption, partial coverage through *Duty Cycling* has been studied as well. In [25], [26], the authors provide a theoretical analysis and simulation on the delay (or stealth distance) before a target is detected. In [25], the Quality of Surveillance (QoSv) is defined as the reciprocal value of the expected travel distance before mobile targets are first detected by any

sensor. In [27], nodes coordinate among each other to guarantee the worst-case detection delay and minimize the average detection delay. In [28], [29], [30], the theoretical foundations for laying barriers with stealthy and wireless sensors are proposed in order to detect the intrusion of mobile targets approaching the barriers from the outside.

The closest related work is *virtual patrol* [31] in which a *virtual patrol* moves along the predefined path in two-dimensional space and triggers sensors adjacent to the virtual patrol's path for detection. This virtual patrol is similar to the concept of our virtual scan. However, the uniqueness of our work can be clearly identified in the following respects: 1) our work focuses on surveillance in road network, where legacy two-dimensional solutions cannot directly apply and 2) we are the first to guarantee target detection with sensing hole handling while sensor network deteriorates.

6 Conclusion

Specially tailored for road networks, this work introduces *VISA* based on the concept of virtual scanning. *VISA* propagates sensing waves along the roadways and detects vehicles entering into the target road network before they reach the protection points. We demonstrate analytically and empirically the feasibility of achieving longer network lifetime and shorter detection delay simultaneously. In addition, we propose an optimal algorithm to deal with the initial sensing holes at the deployment time as well as the sensing holes due to node failure and the heterogeneous energy budget among sensors by optimally labeling additional pseudoprotection or pseudoentrance points. Evaluation shows orders-of-magnitude longer network lifetime than the always-awake method, and as much as 10 times longer than the duty cycling algorithms. We believe that this work opens a promising direction of road network surveillance. Future work includes the following:

1. the perimeter protection of road networks,
2. protection design with bounded detection delay for the quality of surveillance,
3. optimal sensor placement with minimal detection delay, and
4. civil applications, such as the contamination detection in water flow systems.

Appendix A

Average Detection Time in Virtual Scanning

In this section, we derive the Average Detection Time (ADT) for virtual scanning in a road segment. At first, for clarity, we assume that vehicle speed is constant, the same as with the maximum speed v. Later, we relax this assumption, that is, vehicle speed is bounded variable speed.

Enter during scan time. Fig. 24a shows a vehicle enters during the scan time T_{scan}. Since each node covers road segment of length l/n, the virtual scan wave moves along the road segment with the speed $v_{scan} = l/(nw)$. The relative speed between the scan wave and the vehicle is $l/(nw) + v$. Suppose that a vehicle enters at t_a *after* the start of scan, the scan wave has already traveled $lt_a/(nw)$. Therefore, it takes $(l - \frac{lt_a}{nw})/(\frac{l}{nw} + v)$ seconds before the scan wave reaches the vehicle, which is the detection delay D_{scan}. Integrated t_a over the interval $[0, nw]$, expected detection delay (denoted by $E[D_{scan}]$) during scan time is

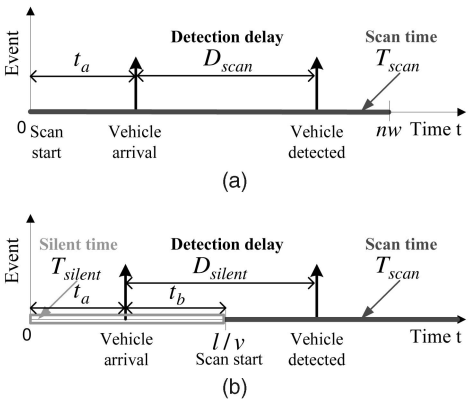

Fig. 24. Vehicle detection cases in virtual scanning. (a) Vehicle detection during scan time. (b) Vehicle detection during silent time.

$$E[D_{scan}] = \int_0^{nw} \frac{nwl - lt_a}{nwv + l} \frac{1}{nw} dt_a = \frac{nwl}{2(nwv + l)}. \quad (6)$$

Enter during silent time. Fig. 24b shows a vehicle enters during the silent time T_{silent}. Suppose that a vehicle enters at t_a *after* the start of silent time. As shown in Fig. 24b, since it enters at t_b *before* the start of scan, the vehicle has already traveled $t_b v$. Therefore, it takes $(l - t_b v)/(\frac{l}{nw} + v)$ seconds before the scan wave reaches the vehicle. For the detection delay, we also need to count the vehicle movement time t_b along with the previous detection delay after the start of the scan. Note that $t_b = l/v - t_a$. Thus, the detection delay becomes $D_{silent} = \frac{l}{v} - t_a + (l - (\frac{l}{v} - t_a)v)/(\frac{l}{nw} + v)$. Integrated t_a over the interval $[0, l/v]$, expected detection delay (denoted by $E[D_{silent}]$) during the silent time is

$$E[D_{silent}] = \int_0^{l/v} \frac{nwl - lt_a + l^2/v}{nwv + l} \frac{v}{l} dt_a = \frac{2nwl + l^2/v}{2(nwv + l)}. \quad (7)$$

By combining both scenarios, we can compute the expected ADT for the virtual scanning as follows:

$$E[D] = \frac{nw}{nw + l/v} E[D_{scan}] + \frac{l/v}{nw + l/v} E[D_{silent}] = \frac{l}{2v}. \quad (8)$$

Appendix B

ADT Computation for Bounded Variable Vehicle Speed

Now we relax the assumption that vehicle speed is constant, the same as with the maximum speed v_{max}. We assume that vehicle speed is bounded variable speed $v = [v_{min}, v_{max}]$ for $0 < v_{min} < v_{max}$. Since this relaxation causes the silent time to be changed as $T_{silent} = l/v_{max}$, the expected detection delay during the silent time becomes as follows:

$$\begin{aligned} E[D_{silent}] &= \int_0^{l/v_{max}} \frac{nwl - lt_a + l^2/v_{max}}{nwv + l} \frac{v_{max}}{l} dt_a \\ &= \frac{2nwl + l^2/v_{max}}{2(nwv + l)}. \end{aligned} \quad (9)$$

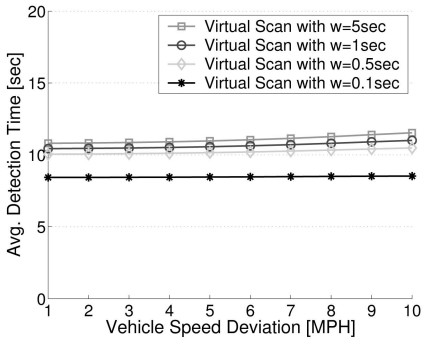

Fig. 25. The impact of vehicle speed deviation on average detection time.

Since there exists no change in the detection delay during the scan time, the combined expected ADT is

$$E[D] = \frac{nw}{nw + l/v_{max}} E[D_{scan}] + \frac{l/v_{max}}{nw + l/v_{max}} E[D_{silent}] = \frac{l(nwv_{max} + l)}{2v_{max}(nwv + l)}. \quad (10)$$

Clearly, (10) becomes the same one as with (8) for $v = v_{max}$.

Now we can compute the average detection time for bounded variable vehicle speed. Suppose that the vehicle speed has a Gaussian distribution with $N(\mu_v, \sigma_v)$ in the range of $v = [v_{min}, v_{max}]$. We can compute the expected ADT for this setting as follows:

$$E[D] = \int_{v_{min}}^{v_{max}} \frac{l(nwv_{max} + l)}{2v_{max}(nwv + l)} \frac{1}{\sqrt{2\pi}\sigma_v} e^{-\frac{(v-\mu_v)^2}{2\sigma_v^2}} dv. \quad (11)$$

As shown in Fig. 25, we can see the trend of ADT according to the vehicle speed deviation in bounded variable speed, where $v_{max} = 70$ MPH, $v_{min} = 10$ MPH, $\mu_v = 40$ MPH, and $\sigma_v = \{1, 2, \ldots, 10\}$ MPH. It can be observed that the ADT tends to slightly increase as the vehicle speed deviation σ_v increases.

ACKNOWLEDGMENTS

This research was supported by the Digital Technology Center at the University of Minnesota, and in part by the US National Science Foundation (NSF) under grant nos. CNS-0626609, CNS-0626614, and CNS-0720465.

REFERENCES

[1] Z. Abrams, A. Goel, and S. Plotkin, "Set K-Cover Algorithms for Energy Efficient Monitoring in Wireless Sensor Networks," *Proc. Int'l Symp. Information Processing in Sensor Network (IPSN),* 2004.

[2] M. Cardei, M.T. Thai, Y. Li, and W. Wu, "Energy-Efficient Target Coverage in Wireless Sensor Networks," *Proc. IEEE INFOCOM,* 2005.

[3] S. Kumar, T.H. Lai, and J. Balogh, "On K-Coverage in a Mostly Sleeping Sensor Network," *Proc. ACM MobiCom,* 2004.

[4] D. Tian and N. Georganas, "A Node Scheduling Scheme for Energy Conservation in Large Wireless Sensor Networks," *Wireless Comm. and Mobile Computing J.,* vol. 3, pp. 271-290, May 2003.

[5] A. Savvides, C.C. Han, and M.B. Srivastava, "Dynamic Fine-Grained Localization in Ad-Hoc Networks of Sensors," *Proc. MobiCom,* July 2001.

[6] J. Elson, L. Girod, and D. Estrin, "Fine-Grained Network Time Synchronization Using Reference Broadcasts," *Proc. Symp. Operating Systems Design and Implementation (OSDI),* Dec. 2002.

[7] M. Maróti, B. Kusy, G. Simon, and Ákos Lédeczi, "The Flooding Time Synchronization Protocol," *Proc. Int'l Conf. Embedded Networked Sensor Systems (Sensys),* Nov. 2004.

[8] L. Gu et al., "Lightweight Detection and Classification for Wireless Sensor Networks in Realistic Environments," *Proc. Int'l Conf. Embedded Networked Sensor Systems (Sensys),* Nov. 2005.

[9] J. Hwang, T. He, and Y. Kim, "Exploring In-Situ Sensing Irregularity in Wireless Sensor Networks," *Proc. Int'l Conf. Embedded Networked Sensor Systems (Sensys),* pp. 289-303, Nov. 2007.

[10] Y. Gu and T. He, "Data Forwarding in Extremely Low Duty-Cycle Sensor Networks with Unreliable Communication Links," *Proc. Int'l Conf. Embedded Networked Sensor Systems (Sensys),* pp. 321-334, Nov. 2007.

[11] G. Lu, N. Sadagopan, B. Krishnamachari, and A. Goel, "Delay Efficient Sleep Scheduling in Wireless Sensor Networks," *Proc. INFOCOM,* 2005.

[12] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein, *Introduction to Algorithms,* second ed. MIT Press, 2003.

[13] Q. Fang, J. Gao, and L.J. Guibas, "Locating and Bypassing Routing Holes in Sensor Networks," *Proc. IEEE INFOCOM,* Mar. 2004.

[14] S. Funke and C. Klein, "Hole Detection or: 'How Much Geometry Hides in Connectivity?'," *Proc. Ann. Symp. Computational Geometry (SoCG),* June 2006.

[15] T. Asano, B. Bhattacharya, M. Keil, and F. Yao, "Clustering Algorithms Based on Minimum and Maximum Spanning Trees," *Proc. Fourth Ann. Symp. Computational Geometry,* 1988.

[16] T. He et al., "An Energy-Efficient Surveillance System Using Wireless Sensor Networks," *Proc. Int'l Conf. Mobile Systems, Applications, and Services (Mobisys),* June 2004.

[17] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M.B. Srivastava, "Coverage Problems in Wireless Ad-Hoc Sensor Networks," *Proc. IEEE INFOCOM,* Apr. 2001.

[18] X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, and C. Gill, "Integrated Coverage and Connectivity Configuration in Wireless Sensor Networks," *Proc. Int'l Conf. Embedded Networked Sensor Systems (Sensys),* Nov. 2003.

[19] S. Shakkottai, R. Srikant, and N. Shroff, "Unreliable Sensor Grids: Coverage, Connectivity, and Diameter," *Proc. IEEE INFOCOM,* Apr. 2003.

[20] H. Zhang and J. Hou, "Maintaining Sensing Coverage and Connectivity in Large Sensor Networks," *Ad Hoc & Sensor Wireless Networks,* vol. 1, Mar. 2005.

[21] F. Ye, G. Zhong, J. Cheng, S. Lu, and L. Zhang, "PEAS: A Robust Energy Conserving Protocol for Long-Lived Sensor Networks," *Proc. Int'l Conf. Distributed Computing Systems (ICDCS),* May 2003.

[22] T. Yan, T. He, and J. Stankovic, "Differentiated Surveillance Service for Sensor Networks," *Proc. Int'l Conf. Embedded Networked Sensor Systems (Sensys),* Nov. 2003.

[23] W. Cheng, M. Li, K. Liu, Y. Liu, X. Li, and X. Liao, "Sweep Coverage with Mobile Sensors," *Proc. Int'l Parallel and Distributed Processing Symp. (IPDPS),* Apr. 2008.

[24] S. Slijepcevic and M. Potkonjak, "Power Efficient Organization of Wireless Sensor Networks," *Proc. Int'l Conf. Comm. (ICC),* 2001.

[25] C. Gui and P. Mohapatra, "Power Conservation and Quality of Surveillance in Target Tracking Sensor Networks," *Proc. ACM MobiCom,* Sept. 2004.

[26] S. Ren, Q. Li, H. Wang, X. Chen, and X. Zhang, "Analyzing Object Tracking Quality under Probabilistic Coverage in Sensor Networks," *ACM Mobile Computing and Comm. Rev.,* vol. 9, no. 1, Jan. 2005.

[27] Q. Cao, T. Abdelzaher, T. He, and J. Stankovic, "Towards Optimal Sleep Scheduling in Sensor Networks for Rare Event Detection," *Proc. Int'l Symp. Information Processing in Sensor Networks (IPSN),* 2005.

[28] S. Kumar, T. Lai, and A. Arora, "Barrier Coverage with Wireless Sensors," *Proc. MobiCom,* Aug. 2005.

[29] A. Chen, S. Kumar, and T.-H. Lai, "Designing Localized Algorithms for Barrier Coverage," *Proc. ACM MobiCom,* Sept. 2007.

[30] P. Balister, B. Bollobás, A. Sarkar, and S. Kumar, "Reliable Density Estimates for Achieving Coverage and Connectivity in Thin Strips of Finite Length," *Proc. ACM MobiCom,* Sept. 2007.

[31] C. Gui and P. Mohapatra, "Virtual Patrol: A New Power Conservation Design for Surveillance Using Sensor Networks," *Proc. Int'l Symp. Information Processing in Sensor Networks (IPSN),* Apr. 2005.

Jaehoon Jeong received the BS degree from the Department of Information Engineering at Sungkyunkwan University in Korea in 1999 and the MS degree from the School of Computer Science and Engineering at Seoul National University in Korea in 2001. He is currently working toward the PhD degree in the Department of Computer Science and Engineering at the University of Minnesota. Also, as a researcher in the Electronics and Telecommunications Research Institute (ETRI), he has participated in the Internet Standardization in the Internet Engineering Task Force (IETF), such as IPv6 DNS Configuration. He has published two IETF standards of RFC 4339 and RFC 5006 for IPv6 DNS Configuration. His current research interests include wireless sensor networking for road networks and the data forwarding in vehicular networks. He is a student member of the IEEE and a member of the ACM.

Yu Gu is currently working toward the PhD degree in the Department of Computer Science at the University of Minnesota. His research interests include wireless sensor networks, real-time computing, and distributed systems. He is the author and coauthor of more than 17 papers in premier journals and conferences. His publications have been selected as graduate-level course materials by more than 10 universities worldwide. He is a student member if the IEEE and a member of the ACM SIAM.

Tian He received the PhD degree from the University of Virginia in 2004, under the supervision of professor John A. Stankovic. He is currently an assistant professor in the Department of Computer Science and Engineering at the University of Minnesota—Twin City. He is the author and coauthor of more than 90 papers in premier sensor network journals and conferences with more than 4,000 citations. His publications have been selected as graduate-level course materials by more than 50 universities in the US and other countries. He has received a number of research awards in the area of sensor networking, including four best paper awards (MSN 2006 and SASN 2006, MASS 2008, and MDM 2009). He is also the recipient of the US National Science Foundation (NSF) CAREER Award 2009 and McKnight Land-Grant Professorship 2009-2011. He served a few program chair positions in international conferences and on many program committees, and also currently serves as an editorial board member for four international journals including the *ACM Transactions on Sensor Networks*. His research interests include wireless sensor networks, intelligent transportation systems, real-time embedded systems, and distributed systems, supported by the NSF and other agencies. He is a member of the ACM and the IEEE.

David H.C. Du received the BS degree in mathematics from National Tsing-Hua University, Taiwan, R.O.C., in 1974, and the MS and PhD degrees in computer science from the University of Washington, Seattle, in 1980 and 1981, respectively. He is currently the Qwest chair professor in the Computer Science and Engineering Department at the University of Minnesota, Minneapolis. His research interests include cyber security, sensor networks, multimedia computing, storage systems, high-speed networking, high-performance computing over clusters of workstations, database design, and CAD for VLSI circuits. He has authored and coauthored more than 210 technical papers, including 100 referred journal publications in his research areas. He has also graduated 49 PhD and 80 MS students. He is currently serving a number of journal editorial boards. He has also served as a guest editor for a number of journals including the *IEEE Computer*, *IEEE*, and *Communications of the ACM*. He has also served as conference chair and program committee chair for several conferences in multimedia, database, and networking areas. Most recently, he served as the general chair for the IEEE Security and Privacy Symposium (Oakland, California) 2009 and program committee cochair for the International Conference on Parallel Processing 2009. He is a fellow of the IEEE and the Minnesota Supercomputer Institute.